COMPUTER AIDED REDUCTION OF GEOTECHNICAL ENGINEERING BORING DATA

By

MARK WALLACE PABST II

Bachelor of Science in Civil Engineering

Oklahoma State University

Stillwater, Oklahoma

1978

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
May, 1981

COMPUTER AIDED REDUCTION OF GEOTECHNICAL

ENGINEERING BORING DATA

Thesis Approved:

James L. Parcher
Thesis Adviser

Donald L. Snethen

T. Allen Haikins

Norman N Durham
Dean of the Graduate College

ACKNOWLEDGMENTS

The author wishes to express his appreciation to Dr. James V. Parcher for his guidance and encouragement during the preparation of this study. The author also wishes to thank Dr. T. A. Haliburton and Dr. D. R. Snethen for their advice and suggestions during the preparation of this manuscript.

The author is indebted to Mueser, Rutledge, Johnston, and DeSimone of New York for providing the geotechnical investigation information from the Providence, Rhode Island site. Special thanks also go to the University Computer Center at Oklahoma State University for providing valuable time saving suggestions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF PLATES

NOMENCLATURE

DE	desired elevation
CBE	closest element of data
ELEV	elevation of the particular element under consideration
DIST	straight line distance from the element at which the elevation is known to the element which is being considered

CHAPTER I

INTRODUCTION

Since the mid-1950's, with the advent of the first generation computers, man has been searching for more ways to have the computer do his repetitious and tedious tasks. Many of these tasks have involved the engineering community.

Presently, computers are becoming so economical that small businesses and even individuals can afford the smaller systems.

As the 1990's approach it appears that the fourth generation of computers will come into being. These machines will be approximately 50 times faster than today's large machines, such as the IBM 370/168. It has been recognized that as computers become more accessible and economical, more engineering problems will be solved by them.

The solution of engineering problems by computers has met with varying degrees of success. Problems in the structural engineering field are found to be much more easily modeled than problems from areas such as soil mechanics. As a comparison, the analysis of material strength may be considered. In the case of structural engineering, the material properties are well-defined and consistent, whereas in the field of soil mechanics the dependability of knowledge of material properties is always a desired goal.

Merritt (7) has written that different branches of Civil Engineering

have a varying range of acceptance to computer-aided solutions. His approximations may be seen in Table I.

TABLE I

POTENTIAL COMPUTER APPLICATIONS IN CIVIL ENGINEERING DESIGN AND CONSTRUCTION

Field	Percent of Design Calculations Applicable to Computer Usage
Highways	95
Surveying	90
Hydraulics	70
Soil Mechanics	40
Stress Analysis	90
Critical Path Method	90

It will be shown that one routine problem of the Soils Engineer may be solved by the computer. This problem considers the analysis of boring data and the reduction of that data so that the engineer may make judgments as to the practicality of certain foundations. Because of certain inherent capabilities of the computer, this information may be presented to the engineer in a way that has not been available previously (3). This capability will enhance the engineer's ability to make sound judgments concerning foundation analysis (12, 14).

CHAPTER II

PREVIOUS WORK

Computers have fulfilled a large role in engineering problem-solving. The Integrated Civil Engineering System (ICES) has been used extensively since its conception in the 1960's and is still widely used today. This system deals with Civil Engineering related problems and it is quite diversified in the type of problems that it may solve. Another widely used program is SYMVU from Harvard University (3). This program will plot data in various three-dimensional forms. Other work has also been done at the government and university level to solve more specific problems. As an example, it is known that many transportation departments across the United States have their own programs which reduce and plot data pertaining to surveying and route design. Similarly, many university graduate students have been given a problem statement in which they are to display three-dimensional data while considering the hidden line problem or some other condition which will make the problem more interesting (9, 10).

One aspect of computer plotting and mapping which is of particular value is the three-dimensional plot. In some instances it may be desired to view a landform three-dimensionally at a time when aerial photography is not available. If this is the case, and a topographic map is available, a computer-plotted map may be constructed. Computer-plotted maps also have the advantage of noting a scale on the plot so that the user

may get an idea of the proportions of the landform. An example of such a plot is shown in Figure 1. This plot was created by the Digital Elevation Model program at the Western Mapping Center of the U.S. Geological Survey.

These examples given an indication of the present state of computer software development in the engineering community. All too often it is the case that a specific problem has already been solved by some agency, yet the program is written in a way that the entry of data is difficult (2, 6). For example, the program will solve only a limited number of problems, or some other shortcoming is present which makes utilization difficult (3).

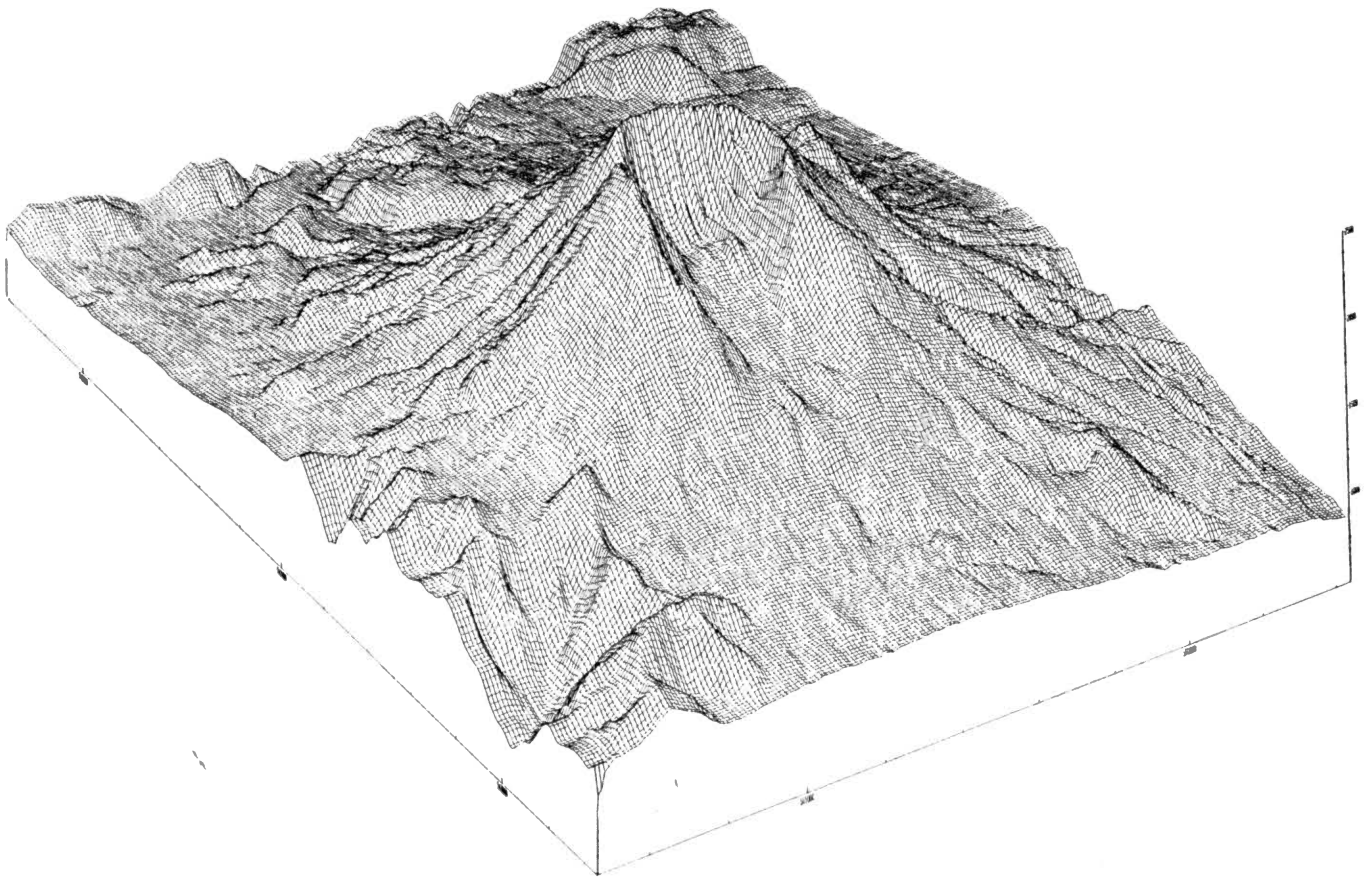


Figure 1. Three Dimension Computer Representation of Mount St. Helens After the May 18, 1980, Eruption

CHAPTER III

IMPLEMENTATION

Several procedures were available to obtain the desired goals of this project. The greatest flexibility was found in the area of choosing a language for program development. Since this problem is of a technical nature, the choice of languages was reduced to Beginner's All-Purpose Symbolic Instruction Code (BASIC), Programming Language I (PL/I), or FORmula TRANslator (FORTRAN). All three languages have shortcomings and these were weighed against one another in order to obtain a program which would be easiest to develop and utilize.

FORTRAN for many years has been a language which has required a set of restrictive rules which are cumbersome on the user in relation to the input of data. At Oklahoma State University both the present versions of PL/I and BASIC will not allow a programmer to write into or call a plot data set. Observing that it would be easier to adjust for the input data deficiency than the plot data set problem, it was decided to develop the program in FORTRAN (1).

There are two common software packages for plotting which are generally available for program development. These are the Calcomp and Complot packages. Because of its versatility, Calcomp was chosen for use in this study.

To correct for the deficiency of FORTRAN in the area of data input, a free format input subroutine was implemented. This utility allows the

programmer to develop his own language for data acquisition. The user is then able to use commands of the input language to enter his data (5, 11, 13).

Module Development

The program to fulfill the desired goals of this study was given the name Geotechnical Engineering Plotting System (GEPS), and will be referred to as such in the remaining chapters.

GEPS was developed using structured programming techniques wherever possible. The concept behind structured programming is to develop a program in a top-down format in which each subroutine or module is executed sequentially. Following is a description of each of these modules and their function.

The program may solve two basic types of problems which have fundamental similarities. The first type of problem is most closely related to analyzing data from boring programs. This module has been named GLOMOD and will call the plan, gridding, cross section, and isometric subroutines. The other type of problem that may be solved by the program entails the plotting of three-dimensional data in an isometric map. These data could be taken from topographic maps or mathematical equations in three unknowns. This module is named FIEMOD and calls the plan and isometric subroutines.

Methods of Analysis

Both the GLOMOD and FIEMOD modules are capable of taking input data in English or metric units. Results are then given in both systems or units.

The location plan subroutine will convert the given input units to the units of the opposing system. The program will also plot a location plan which will give the latitude and longitude values on the axes of the map and plot the data point at the correct location along with its name.

Cross sections specified by the user will be analyzed by the program. The program will list the borings that the user has supplied for a particular cross section, the east-west (latitude) distance, north-south (longitude) distance, and the straight-line distance. The plot of these cross sections will show the boring with its name, the even ten-foot elevation interval, and the soil strata interface line. The soil strata name will not appear on the cross section; therefore, it is important to enter the soil strata types in the correct order. The program requires that these data be entered in a top-down manner. The vertical limits of the cross section is always scaled into a 5½ inch space. Thus if the total cross section depth was 5 feet, the scale would be one inch equals 10 feet. The horizontal scale is always one-quarter as large as the vertical scale; therefore, from the previous example it would be understood that the horizontal scale would be one inch equals 40 feet.

The program will also draw a three-dimensional view of each strata type that has been supplied by the user. Other information supplied by the user consists of the angle of view of the site, the elevation in degrees from which the plot is viewed, the number of lines used to create the plot, and the separation of points which are to be examined. The program will respond by informing the user of the boundary coordinates for a particular plot and the maximum pen displacement of the plot. The plotter output will inform the user of the orientation of the plot and which soil stratum is being viewed.

A gridding function is called by the GLOMOD module to inform the user of the approximate integer elevations of the data points. The gridding routine will take the plan area and dissect it into a 25 x 25 matrix. At each node the system will determine the elevation of a particular stratum to the nearest one-foot interval. A corresponding legend is given to aid the user in locating the proper coordinates of a particular node.

The program also has error analysis capabilities for input data. The format for data entry lent itself to easy error analysis and the topic of data entry will be discussed later.

The error analysis procedures will tell the user specifically, in easy to understand terms, if he has entered data out of sequence or if a data item has been mistyped. Since the format for data entry is slightly different between the GLOMOD and FIEMOD routines, error detection subprograms were developed for each. Once an error has been detected in the input stream, execution is halted by the program.

For the program to develop the isometric maps, some way must be devised to determine points of elevation over the entire site. Since the points of elevation are known only in the borings, a method must be developed to determine the elevation of the points between the borings. There are two techniques which are commonly used. These are the algebraic and the interpolation algorithms. The algebraic form will take the desired point, find the surrounding borings with their corresponding values of elevation, determine all of the surrounding slopes, and then extrapolate the elevation of the desired point from these slopes. The interpolation method will take the desired point, find a prescribed number of surrounding borings and interpolate the desired elevation by a weighting factor determined by the distance of the point from a particular boring. The

weighting factor is usually the square of the distance between the points. Of the two methods, the algebraic system is more accurate, but more cumbersome to develop. In the writer's opinion, the increased effort to develop the algebraic method does not pay off in the slight increase in accuracy, especially when considering the integrity of the input data. Therefore, the interpolation method was utilized, and will be discussed presently.

The algorithm is manipulated by the equation of the form:

$$DE = \frac{1}{2}\left[CBE + \sum_{n=1}^{9} \frac{\frac{elev}{dist^2}}{\frac{1}{dist^2}}\right]$$

Notice that this form of the equation considers the nine borings nearest to the point under inspection. If there are less than nine points, then that number will be used. It has been stated that the range of 4 to 12 borings will give the optimum results. Analyzing the 4 nearest borings will substantially reduce the time required for the computer to solve the problem but will give an unrefined view of the data. On the other hand, considering 10 or more borings will begin to exceed the practical accuracy of the system and substantially increase the execution time.

The major deficiency with the interpolation functions is that the interpolated values will always be less than the maximum values of the set being considered. This is disadvantageous when considering boundary problems that occur in mapping situations. In reality, if the region were a hill, the interpolation algorithms would never detect it because the values would be smaller than the maximum value in the boring cluster.

The algebraic form does not have this shortcoming. That is, the algebraic method will extrapolate the positive slopes and correctly create the data denoting a hill (4, 6).

The program uses the Complot compatible subroutines to create the plot data set. These subroutines are found in most large systems and are capable of creating several different forms of hardcopy. The need for obtaining different forms of hardcopy is dependent on the resolution requirements of the user. A low resolution plot may be obtained from a line printer; a plot of medium resolution may be obtained by viewing a graphics terminal; and a high resolution plot may be obtained from a pen plotter. The examples given in this study were produced on a pen plotter with a resolution of 0.05 inches. This means that the shortest line that can be drawn is 0.05 inches long. The program may be given the desired computational resolution by the user. Thus, the user may reduce the cost of the program run by specifying a computational resolution of 0.25 inches. The smallest resolution that may be entered is 0.01. Notice that greater precision may be obtained on the computational level than the plotting level. Care should be taken when specifying the resolution requirements of a problem because of the increase in execution time. That is, to double the resolution of a plot may require ten times as much execution time.

A sorting method is also utilized to increase the efficiency of the system. Recall that the interpolation function will search for the nine nearest points and then determine the magnitude of the intermittent point. In order to find the nine nearest borings, the system opens a net and then counts the number of borings in the net. If less than nine points are obtained, the net is enlarged. Upon enlargement the possibility exists that more than nine points would fall inside the net. If this is

the case, the borings are sorted according to distance from the point in question and as soon as the nine nearest borings have been utilized the routine is halted (4, 6).

The Problem Oriented Languages

SCAN is a set of FORTRAN subroutines which allows the program developer to write his own language for data entry. This type of language is commonly known as a Problem Oriented Language (POL) (5). The advantage of POL's over standard FORTRAN input is the ability to enter data in more English-like statements and the availability of free format data placement. A POL was written for each of the two routines, GLOMOD and FIEMOD, and are discussed below.

Since the GLOMOD portion of the system deals with Geotechnical Engineering aspects of problem solving, its POL was developed in a way to make data input easiest for the Geotechnical Engineer. This language can be broken into several segments which may be seen in Table II.

TABLE II

HIERARCHY OF GLOMOD KEYWORDS

Command	Reference	Specification
Engage	Hole	Separation
Grid/Nogrid	Elevation	Lines
Drop/Nodrop	Latitude	Viewing
Solid/Nosolid	Longitude	Elevation
Move/Nomove	Depth	Angle
Imperial		Number of
Metric		Borings
		Sections
		Strata

The command portion is used to drive the system. The user specifies his command word to obtain the desired characteristics of a program run. The results of these commands are given in Table III.

TABLE III

GLOMOD POL COMMAND FUNCTIONS

Keyword	Function
Engage GLOMOD	Execute the GLOMOD module
Grid	Construct a gridded surface on the three-dimensional views
Nogrid	Construct surficial line in one direction
Drop	Construct lines from three-dimensional surface to boundary block
Nodrop	Suppress drop function
Solid	Construct boundary block
Nosolid	Suppress boundary block
Move	Moves pen off of three-dimensional view
Nomove	Constructs one three-dimensional view on top of another
Imperial	Input units will be in English units
Metric	Input units will be in metric units

The reference keywords are used to indicate to the program which data types follow. The word HOLE will specify the name for the boring; ELEVATION, the elevation of the boring; LATITUDE, the east-west coordinate of the boring; LONGITUDE, the north-south coordinate of the boring; DEPTH, the total depth of the boring.

Keywords of the specification type will define numerical quantities to drive the system. The functions of the specification keywords are given in Table IV.

TABLE IV

GLOMOD POL SPECIFICATION FUNCTIONS

Keyword	Function
Separation Lines	Computational resolution of system Number of lines used to construct three-dimensional plot
Viewing Elevation	Angle in degrees of elevation that the three-dimensional plots will be viewed
Viewing Angle	Angle in degrees of the adjusted azimuth
Number of Borings	Total number of borings in the problem
Number of Sections	Total number of cross sections that will be requested by the user
Number of Strata	Total number of strata in this problem, including the water table

Figure 2 is an illustration to aid the user when using the VIEWING ANGLE specification. Notice that the graph is similar to the azimuth technique used in surveying.

The FIEMOD POL contains an abbreviated set of the GLOMOD commands. The hierarchy for the FIEMOD keywords may be found in Table V.

The commands which are common to both GLOMOD and FIEMOD may be found in Table III. The reference keywords of ELEVATION, LATITUDE, and LONGITUDE are identical to their GLOMOD counterparts. In the FIEMOD system

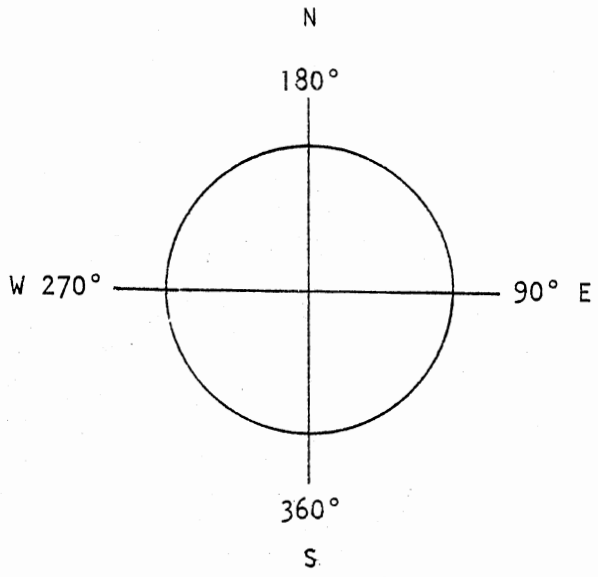


Figure 2. Isometric View Orientation

TABLE V

HIERARCHY OF FIEMOD KEYWORDS

Command	Reference	Specifications
Engage	Point	Separation Lines
Grid/Nogrid	Elevation	Viewing Angle Elevation
Drop/Nodrop	Latitude	
Solid/Nosolid	Longitude	
Move/Nomove	Area	
Imperial		
Metric		
Start		
Continue		
End		

the reference keyword POINT is an alias for HOLE, while the new keyword AREA denotes that the name for the present data set follows. All of the FIEMOD specification keywords are identical to the GLOMOD specifications and are listed in Table VI (5, 11, 13).

TABLE VI

SELECT FIEMOD POL FUNCTIONS

Keyword	Function
Engage FIEMOD	Execute the FIEMOD module
Start	Indicates the beginning of a data set for a problem
Continue	Indicates another problem is to follow along with its data
End	Indicates the end of data for the program run

CHAPTER IV

DISCUSSION OF RESULTS

The computer listing for the program and examples may be found in the appendices. Appendix A contains the copy of the program source listing. Appendices B, C, and D contain the output listing from the example problems. In the packet at the rear of this study are Plates 1 through 14. Following is a discussion of the results presented in these appendices.

Listings

Appendix B contains the output listing for the GLOMOD program run. The data for this run were obtained from the boring program performed at the site for the future Providence, Rhode Island, Railroad Station and Parking Garage (8). This appendix contains the numerical breakdown of the various subroutines.

The first page of the output listing consists of the GEPS banner page. Following the first page is the echo print of the input data. Successful execution of this input data is denoted by the "END OF INPUT EXECUTION" message.

The next listing produced by the program displays the "RUN ANALYSIS." The information contained here will help the user in determining to what extent he has used the program's resources. The line which discusses maximum space indicates the portion of array area that was utilized for

that particular problem. The lines which discuss plot time and plot length were developed to assist the system operator. Typically the operator will want an estimate from the user of the size parameters of a particular plot.

The next subject that is dealt with in the output listing is the data produced by the boring location plan routine. In this listing the program displays the boring name, the coordinates of the boring as latitude and longitude, and the elevation of the boring. The coordinate distances and elevations are given in both English and metric units of feet and kilometers, respectively.

Following the boring location plan output comes the cross-sectioning information. The first portion of this listing informs the user of the name that was given to the boring, the name the program gave to the boring, the total depth of the boring in English and metric units, and the number of strata encountered by that boring. If water table readings were taken in a boring, then the system considers the water table to be a stratum and includes it in the number of soils category. The second segment of the cross sections listing exhibits each of the soil strata. In the exhibit for a particular stratum, the borings that encountered that stratum are given. The information that is passed along with each boring is given in the following three columns. The first column indicates the depth to the bottom of that stratum from the ground surface or the reference used as the top of boring, i.e., barge elevation. The next column deals with the thickness of the stratum that is now being analyzed. The final column deals with the elevation of the bottom of stratum being considered in a particular boring. That is, the elevation of the bottom of a stratum equals the elevation of the boring minus the depth to the bottom

of the stratum. The data presented in these three columns are given in both English and metric units of feet and meters. The final listing related to the cross-section subroutine deals with the user specified cross sections. In each of the cross sections, the specified borings are listed along with the calculated displacements from one boring to the next. These displacements are presented in three ways. They are: the difference in latitude or east-west difference, longitude or north-south difference, and the straight line distance, which is the square root of the sum of the squares of the latitude and longitude calculations. As with the previous data, these values are also given in units of feet and kilometers.

The three-dimensional plot listing is found next for each of the soil strata. Information contained in this listing consists of the stratum type, the boundary coordinates for the borings which penetrated the stratum under consideration, an echo print of user specified commands, and the pen displacement for that stratum plot. Three-dimensional plots which are created and exceed a pen displacement of 20 inches set off a warning to the user. This warning suggests that the user check his data for possible errors.

The final segment of listing for the GLOMOD module deals with the contour grid for each stratum. The stratum area is dissected into a 25 x 25 matrix which has been given the axis titles of "nodes" and "points." In the body of the matrix is given the 725 coordinate elevations. These elevations are truncated real values calculated by the interpolation function. Following each contour grid is a coordinate legend. The legend references the nodes to the latitude distances and the points to the longitude distances. These distances are also truncated values which have

units of feet. Upon a normal exit from the GLOMOD module, a "SUCCESSFUL RUN" message is printed.

The listing from the FIEMOD module may be found in Appendix C. It may be noticed that the FIEMOD listing is an abbreviated form of the GLOMOD listing and the previous explanation of the GLOMOD listing also deals with the FIEMOD listing. The FIEMOD listing contains the banner page, input echo print, boring location plan listing, and the three-dimensional plot output. There is no end of processing flag for the FIEMOD routine.

A special program run of the FIEMOD module was made in order to illustrate the error detecting capabilities of the program. This listing may be found in Appendix D.

As with the previous examples, the first page displays the banner output. Following the banner page is the echo print of the input data. Notice that the program only read in the first 48 of the 49 points. This is because an error was detected in "P 48." The program then prints the applicable error message and then cancels execution. Upon inspecting the error message, it is noticed that the error occurred in the value following the keyword "LATITUDE." Upon examining the data statement, it is noticed that the letter "O" has been entered instead of the number "0."

Plates

Note: At the time of the plotting of the plates used in this study, the University Computer Center was experiencing difficulty with the DP-8 plotter. Subsequently, some of the more intricate plotwork was displaced into random locations.

The boring location plan for the GLOMOD routine is illustrated on Plate 1. The location of the boring is drawn with a special symbol and

the borings user specified name is written in the upper right-hand corner of this symbol. On the periphery of the map is given the scale for latitude and longitude in units of kilometers. A north arrow is also constructed for orientation. Plates 2 through 6 give the cross sections which were created by GLOMOD. The cross section name is written in the lower left-hand corner. The vertical scale is noted on the left side of the particular cross section and black lines are drawn horizontally for reference. The horizontal scale is always one-quarter of the vertical scale. The borings are drawn in blue and the name is written at the top of the boring. The red lines denote the interface between different strata types. If the interface lines do not continue to the next boring, this would indicate that the boring did not encounter the soil type found below the interface line. The three-dimensional isometric views produced by the GLOMOD problem set are illustrated in Plates 7 through 12. These plates contain the stratum name, the direction from which the site is being viewed, and a warning that the plot is created by interpolated data.

Plates 13 and 14 were produced by the FIEMOD problem set. Plate 13 is similar to the boring location plan plot discussed above, and Plate 14 is similar to the three-dimensional view isometric plots. The only difference in the plots comes from the difference in the input data.

CHAPTER V

SUMMARY AND CONCLUSIONS

In the course of work, the Geotechnical Engineer must make decisions as to the most appropriate foundation type for a particular site. Site characteristics play an important role in deciding on the most suitable foundation.

For many years engineers have used boring data in order to determine subsurface conditions. More recently, aerial photography has played an increasingly important role in deciphering site conditions. Subsequently, it would be of interest to the engineer to obtain photogrammetric-like information of the soil strata below the ground surface. This may be done by using the plotting capabilities of digital computers.

It has been shown by the program presented in this study that three-dimensional plots may be constructed by a computer with relevant results. The more conventional information such as boring location plans and cross sections are also presented.

Future Work

Effort may be most fruitfully used in reducing the size and execution time of this program. The most expeditious way to attain this goal would come from structuring the system in a new format and writing the program in a different language. Presently, the program becomes bulky when solving even an average size problem. If the system does not per-

perform satisfactorily in a research environment, success in the business community is doubtful.

The author believes that the best way to attain the two goals mentioned above would be with the implementation of the programming language PASCAL. PASCAL, a relatively new programming language, was developed in 1970, and overcomes many of the shortcomings of languages previously mentioned. Its greatest advantages are:

1. A small compiler, which allows it to be placed on mini-computers.
2. Executes approximately twice as fast as FORTRAN.
3. Deals with files in an efficient manner.

If programs are to be developed for use in industry, they must be written in a language which is available in the business community. Since most businesses now have or would most likely purchase a mini-computer, PASCAL has definite advantages over other languages, since these other languages would not fit onto the smaller machines.

Other improvements may be made in the output capabilities of the program. It would be relatively simple to add appurtenances to the input data. These appurtenances would then be plotted on the cross section and location plan plots. Effort may also be wisely used in determining such problem characteristics as boring influence area, areal extent for each stratum, and the center of each stratum area in reference to the center of the site.

A SELECTED BIBLIOGRAPHY

(1) Alexander, D. E., and A. C. Messer. FORTRAN IV Pocket handbook. New York: McGraw-Hill, 1972.

(2) Bowles, J. E. Analytical and Computer Methods in Foundation Engineering. New York: McGraw-Hill, 1974.

(3) Dougenik, J. A., and D. E. Shechan. SYMAP User's Reference Manual. Cambridge: President and Fellows of Harvard College, 1975.

(4) Davis, J. C., and M. K. McCullagh. Display and Analysis of Spatial Data. London: John Wiley and Sons, 1975.

(5) Lopez, L. A. SCAN User's Manual. Champagne-Urbana, Ill.: University of Illinois, Civil Engineering Systems Laboratory, September, 1976.

(6) MacDougall, B. C. Computer Programming for Spatial Problems. London: William Clowes and Sons, Inc., 1976.

(7) Merritt, F. S. Standard Handbook for Civil Engineers. New York: McGraw-Hill, 1976.

(8) Geotechnical Investigation for Proposed Providence Station, Parking Garage and Track Realignment. Report No. NEC-213S. New York: Mueser, Rutledge, Johnston, and DeSimone, April 15, 1980.

(9) Oines, R. "PLOTZ" University Computer Center Publication. Stillwater, Okla.: Oklahoma State University Computer Center, April, 1973.

(10) Pabst, M. W. "Computer Aided Anlaysis of the Providence, Rhode Island Railroad Station Boring Program." Unpublished report. Oklahoma State University, Stillwater, Okla., 1980.

(11) Pabst, M. W. SCAN Guide. Stillwater, Okla.: Oklahoma State University Computer Center, December, 1980.

(12) Pleck, R. B., W. E. Hanson, and T. H. Thornburn. Foundation Engineering. New York: John Wiley and Sons, Inc., 1974.

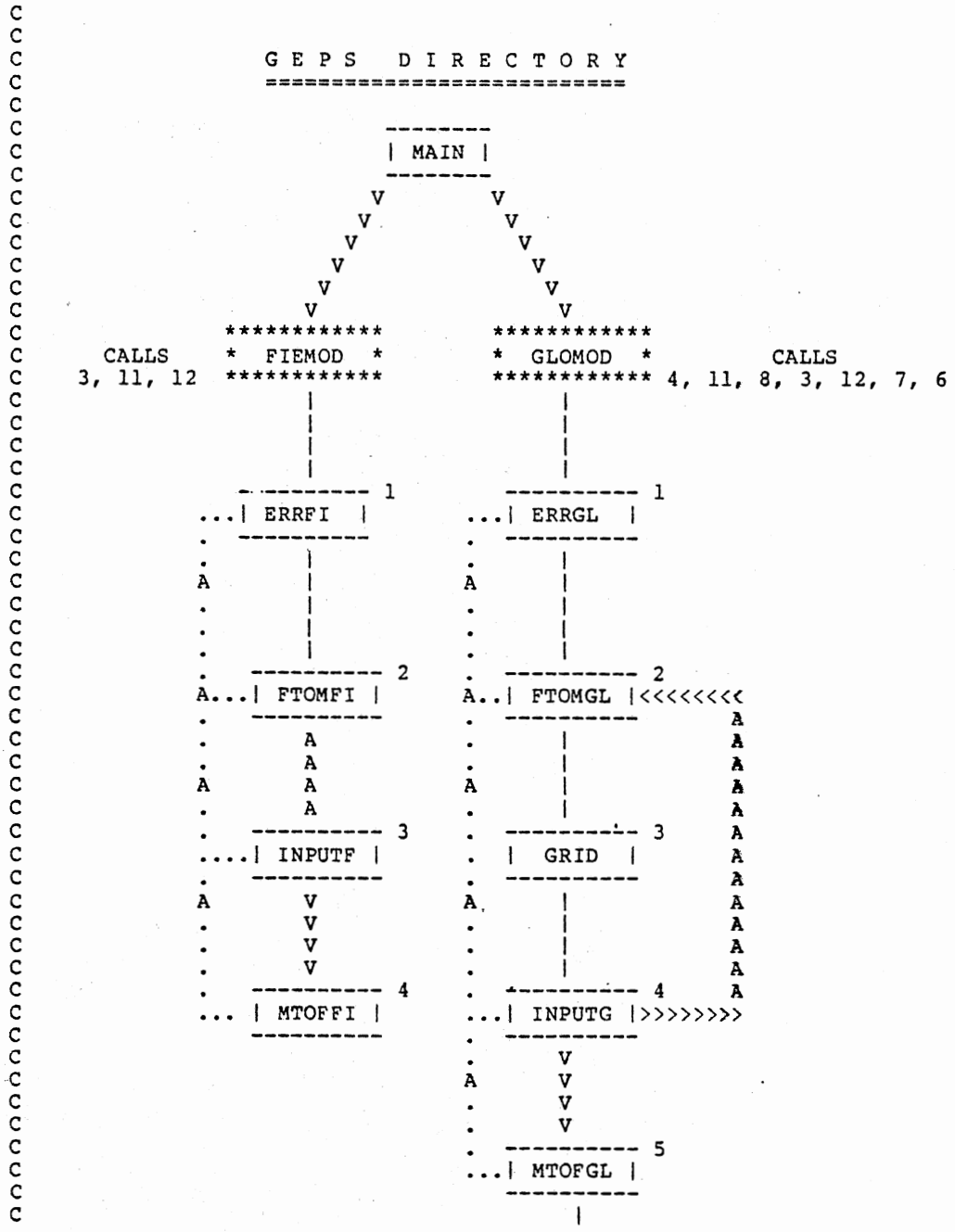
(13) Rehak, D. R. SCAN: A Tool for Translating Problem Oriented Language. Champagne-Urbana, Ill.: University of Illinois, Department of Civil Engineering, n.d.

(14) Tschebotarioff, G. P. Foundations, Retaining and Earth Structures. New York: McGraw-Hill, 1973.

APPENDIX A

PROGRAM SOURCE LISTING

```
C
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$                                                                   $$
C$$                                                                   $$
C$$                              G E P S                              $$
C$$                              =======                              $$
C$$                                                                   $$
C$$                     MARK PABST                                    $$
C$$                     SCHOOL OF CIVIL ENGINEERING                   $$
C$$                     OKLAHOMA STATE UNIVERSITY                     $$
C$$                                                                   $$
C$$   THE GEOTECHNICAL ENGINEERING PLOTTING SYSTEM WAS PROGRAMMED     $$
C$$   IN AN EFFORT TO SUBSTANTIATE A DEFENSE FOR THE AUTHORS          $$
C$$   MASTERS THESIS.                                                 $$
C$$                                                                   $$
C$$   ADDITIONAL INFORMATION ABOUT GEPS MAY BE OBTAINED FROM          $$
C$$   THE THESIS - PABST, M. W. (1981)                                $$
C$$                                                                   $$
C$$   SUBROUTINES PZOSU,CHECK,SET AND LINX WERE DEVELOPED             $$
C$$   AT THE UNIVERSITY COMPUTER CENTER OF OKLAHOMA STATE UNIVERSITY  $$
C$$   AND ALSO ADAPTED TO GEPS.                                       $$
C$$                                                                   $$
C$$   SCAN IS A FREE FORMAT FORTRAN SUBROUTINE DELELOPED AT THE       $$
C$$   UNIVERSITY OF ILLINOIS AT CHAMPAGNE - URBANA.                   $$
C$$                                                                   $$
C$$                                                                   $$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
C
C
C
C
C
C#######################################################################
C##                                                                   ##
C##                                                                   ##
C##   GEPS IS SYSTEM DEPENDENT ON A FOUR BIT WORD MACHINE. ( IBM -    ##
C##   360/370,RCA 70-B).                                              ##
C##                                                                   ##
C##   GEPS UTILIZES CALCOMP SOFTWARE WHICH HAS BEEN ADAPTED TO        ##
C##   COMPLOT HARDWARE.  THE SYSTEM HAS BEEN DEVELOPED ON A DP-8      ##
C##   DRUM PLOTTER WITH 18 INCH PAPER.                                ##
C##                                                                   ##
C##   THE CALCOMP SOFTWARE ALSO SUPPORTS TEKTRONIX PREVIEWING         ##
C##   SYSTEMS.                                                        ##
C##                                                                   ##
C##                                                                   ##
C#######################################################################
C
```


GEPS DIRECTORY
MAIN
CALLS 3, 11, 12
FIEMOD
GLOMOD
CALLS 4, 11, 8, 3, 12, 7, 6
ERRFI
ERRGL
FTOMFI
FTOMGL
INPUTF
GRID
MTOFFI
INPUTG
MTOFGL
1
2
3
4
5

```
C                                          |
C                                          |
C                                          |
C                                    ---------- 6
C  NOTE -                            | STACK  |
C  THE FIELDS NETWORK MODULE         ----------
C  IS PLACED AFTER THE GLOBAL              |
C  NETWORK MODULE.                         |
C                                          |
C  THE CUMMULATIVE NEST THEN               |
C  FOLLOWS THE FIELDS MODULE         ---------- 7
C                                    | THREED |
C                                    ----------
C                                          |
C                                          |
C                                          |
C                                          |
C                                    ---------- 8
C                                    |  XSEC  |
C                                    ----------
C
C
C
C                +++++++++++++++
C               +    N E S T    +
C                +++++++++++++++
C                       |
C                       |                C A L L S
C                A ----------  9
C                  | CHECK  |               B
C                  ----------
C                       |
C                       |
C                B ---------- 10
C                  |  LINX  |
C                  ----------
C                       |
C                       |
C                C ---------- 11
C                  |  PLAN  |
C                  ----------
C                       |
C                       |
C                D ---------- 12
C                  | PZOSU  |               E, F, A
C                  ----------
C                       |
C                E ---------- 13
C                  |  SET   |               F
C                  ----------
C                       |
C                       |
C                F ---------- 14
C                  |  ZED   |               G
```

```
C                  ----------
C                       |
C                       |
C                G ---------- 15
C                  |  ZEE   |
C                  ----------
C
C
C
C*************************************************************
C*                                                           *
C*                                                           *
C*                      MAIN PROGRAM                         *
C*                                                           *
C* THE MAIN PROGRAM PRINTS OUT THE GEPS HEADER AND           *
C* DISCERNS BETWEEN THE GLOBAL AND THE FIELDS NETWORKS.      *
C*                                                           *
C*************************************************************
C
C
C
C GEPS ASSUMES THAT PEN 1 IS BLACK
C                   PEN 2 IS RED
C                   PEN 3 IS BLUE
C
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEXT,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      INTEGER ENG, GLO, FIE
      DATA ENG/'ENG'/, GLO/'GLO'/, FIE/'FIE'/
C
C WRITE OUT THE BANNER PAGE
C
      WRITE(LP,1050)
 1050 FORMAT(1H1,9(/),5(25X,80('M'),/),
     124X,9('M'),3(10X,8('M')), 10X,9('M')/,
     223X,10('M'),3(10X,8('M')),10X,10('M')/,
     X22X,11('M'),2(2X,16('M')),2X,6('M'),2X,8('M'),2X,19('M')/,
     321X,12('M'),2(2X,16('M')),2X,6('M'),2X,8('M'),2X,20('M')/,
     420X,13('M'),2X,16('M'),8X,10('M'),10X,8('M'),10X,13('M')/,21X,
     512('M'),2X,4('M'),4X,8('M'),8X,10('M'),10X,8('M'),10X,12('M')/,
     622X,11('M'),2X,4('M'),4X,8('M'),2X,16('M'),2X,24('M'),2X,11('M')/,
     723X,10('M'),2X,6('M'),2X,8('M'),2X,16('M'),2X,24('M'),2X,10('M')/,
     824X,9('M'),2X,6('M'),2X,8('M'),2X,16('M'),2X,24('M'),2X,9('M')/,
     92(25X,8('M'),10X,8('M'),10X,8('M'),2X,16('M'),10X,8('M')/),
     L5(25X,80('M')/))
      WRITE(LP,50)
 50   FORMAT(5(/),40X,50('*')/,3(40X,'*',48X,'*'/),40X,'*',12X,
     1 'GEOTECHNICAL',24X,'*',/,40X,'*',48X,'*',/,40X,'*',20X,
     2 'ENGINEERING',17X,'*',/,40X,'*',48X,'*',/,40X,'*',27X,
```

```
     3 'PLOTTING',13X,'*',/,40X,'*',48X,'*',/,40X,'*',31X,
     4 'SYSTEM',11X,'*',/,2(40X,'*',48X,'*',/),40X,'*',20X,
     5 'OKLAHOMA',20X,'*',/,40X,'*',22X,'STATE',21X,'*',/,40X,'*',20X,
     6 'UNIVERSTY',19X,'*',/,40X,'*',48X,'*',/,40X,'*',20X,
     7 'LEVEL II',20X,'*',/,40X,'*',48X,'*',/,40X,'*',20X,'MAY 1981',
     8 20X,'*',/,2(40X,'*',48X,'*',/),40X,50('*'))
      WRITE(LP,10)
10    FORMAT(1H1)
      C A L L   R E A D S C
      IF (MATCH(ENG,3)) GO TO 20
      CALL ERRGL(37)
20    IF (MATCH(GLO,3)) CALL GLOBAL
      IF (NXX.EQ.69) GO TO 202
      IF (MATCH(FIE,3)) CALL FIELDS
      IF (IFLIP1.NE.0) GO TO 202
      CALL ERRGL(29)
 202  WRITE(LP,203)
 203  FORMAT(///,5X,'SUCCESFUL RUN.  CONGRADULATIONS  -  HAVE A BEER. ')
      S T O P
      E N D
      SUBROUTINE GLOBAL
C*********************************************************************
C*                                                                   *
C*    G L O M O D                                                    *
C*                                                                   *
C*********************************************************************
C **
C **
C **
C **
C **
C **
C **
C **
C **
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
C
C THE FOLLOWING COMMON BLOCK MOVES DATA FROM SCAN TO GEPS
C
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEXT,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      INTEGER * 4 COM,STAT,SEPE
      DATA COM/'COM'/,NOG/'NOG'/,
```

```
     A NOD/'NOD'/,NOS/'NOS'/,NOM/'NOM'/,STAT/'STAT'/,
     B SEPE/'SEP'/,LINE/'LINE'/
      PRBDEP=0.0
10    C A L L   R E A D S C
      IF (MATCH(COM,3)) GO TO 150
      IF (MATCH(STAT,4)) GO TO 1000
      CALL ERRGL(11)
150   IF (MATCH(NOG,3)) GO TO 250
      IX2 = 9
      GO TO 300
250   IX2 = -9
300   IF (MATCH(NOD,3)) GO TO 350
      IX3 = 9
      GO TO 400
350   IX3 = -9
400   IF (MATCH(NOS,3)) GO TO 450
      IX4 = 9
      GO TO 500
450   IX4 = -9
      IF (MATCH(NOM,3)) GO TO 550
500   IX5 = 9
      IX6 = 9
      GO TO 10
550   IX5 = -9
      IX6 = 9
      GO TO 10
1000  IF (MATCH(SEPE,3)) GO TO 110
130   IF (MATCH(LINE,4)) GO TO 120
      CALL ERRGL(12)
110   IF (NUMR(D55)) GO TO 130
120   IF (NUMI(IX1)) GO TO 140
C
C READ IN ALL INPUT
C
140   CALL INPUTG
C
C CALCULATE THE RUN ANALYSIS
C
      NUSAGE=(NOBOR+NOXSEC+NOSTRA)/0.70
      WRITE(LP,35)
      WRITE(LP,45)
      WRITE(LP,65) NUSAGE
      LENGTH=12+NOXSEC*16+NOSTRA*18
      WRITE(LP,4025) LENGTH
      DO 4000 IPD=1,NOBOR
      PRBDEP=TOTDEP(IPD)+PRBDEP
4000  C O N T I N U E
      WRITE(LP,4005) PRBDEP
      PLOTIM=NOBOR/50.0*10.0+NOXSEC*3.0+(12.*NOSTRA)*IX1/100
      MINTIM=PLOTIM-5.0
      MAXTIM=PLOTIM+5.0
      WRITE(LP,4010) MINTIM,MAXTIM
C
C OPEN THE PLOT DATA SET
```

```
C
C IF THE USER DOES NOT WANT TO CREATE THE PLOT DATA SET, CHANGE THE
C DISPOSITION PARAMETER IN THE GO.FT99 DATA DEFINITION CARD TO
C (NEW,DELETE) OR (OLD,DELETE), WHICHEVER IS APPLICABLE.
C
      CALL PLOTS
C
C CALL THE BORING LOCATION PLAN SUBROUTINE
C
      CALL PLAN
C
C CALL THE CROSS SECTION SUBROUTINE
C
      CALL XSEC
C
C PRINT THE GRID CONTOUR MAP FOR EACH STRATA TYPE
C
C
C DRAW THE ISOMETRIC VIEW OF THE GROUND SUFACE
C
C
C CALCULATE THE PERIMETER OF THE SITE
C
      ZMAX=0.0
      ZMIN=1.0E10
            DO 30 M=1,NOBOR
            IF (XCOORD(M).LT.XMIN) XMIN=XCOORD(M)
            IF (XCOORD(M).GT.XMAX) XMAX=XCOORD(M)
            IF (YCOORD(M).LT.YMIN) YMIN=YCOORD(M)
            IF (YCOORD(M).GT.YMAX) YMAX=YCOORD(M)
            IF (ZGIVEN(M).LT.ZMIN) ZMIN=ZGIVEN(M)
            IF (ZGIVEN(M).GT.ZMAX) ZMAX=ZGIVEN(M)
 30         C O N T I N U E
       FACTO=(XMAX-XMIN)/(ZMAX-ZMIN)
       DIFFX=XMAX-XMIN
       DIFFY=YMAX-YMIN
       D11=XMIN
       D22=XMAX
       D33=YMIN
       D44=YMAX
       E11=ELEVA
       E22=ZANG
      CALL PZOSU(D11,D22,D33,D44,D55,IX1,IX2,IX3,IX4,IX5,IX6,E11,E22)
C
C DRAW THE ISOMETRIC VIEWS OF THE STRATA INTERFACES
C
      CALL THREED(D55,IX1,IX2,IX3,IX4,IX5,IX6)
C
      CALL GRID
C
  35  FORMAT(1H1)
 45   FORMAT(10(/),50X,'<<<<<<<  RUN ANALYSIS  >>>>>>>',///)
 65   FORMAT(42X,'THIS PROBLEM FILLED ',I3,'% OF GEPS MAXIMUM SPACE.',/)
 4005 FORMAT(/,38X,'THIS PROBLEM HAD A TOTAL BORING FOOTAGE OF ',
```

```
     A F6.1,' FEET.',/)
 4010 FORMAT(/,36X,'ESTIMATED TIME TO CONSTRUCT THE PLOT IS ',
     A I3,' TO ',I3,' MINUTES.',/)
 4025 FORMAT(/,40X,'THE APPROXIMATE LENGTH OF THE PLOT IS ',
     A I3,' INCHES.',/)
      R E T U R N
      E N D
C ***************************************************************************
C *                                                                         *
C *    ERRGL                                                                *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE ERRGL(NUMERR)
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      WRITE(LP,1000)
1000  FORMAT(1H1,/,2(5X,25('*'),/),2(5X,2('*'),21X,2('*'),/),5X,2('*'),
     A  8X,'INPUT',8X,2('*'),/,5X,2('*'),8X,'ERROR',8X,2('*'),/,
     B  2(5X,2('*'),21X,2('*'),/),2(5X,25('*'),/))
           GO TO(10,20,30,40,50,60,70,80,90,100,110,120,130,140,
     A  150,160,170,180,190,200,210,220,230,240,250,260,270,280,
     B  290,300,310,320,330,340,350,360,370), NUMERR
      R E T U R N
10    WRITE(LP,11)
11    FORMAT(//,5X,'E R R O R  1',//,
     A  5X,'EXPECTING HOLE OR H, BUT NOT FOUND')
      S T O P
20    WRITE(LP,21)
21    FORMAT(//,5X,'E R R O R  2',//,
     B  5X,'EXPECTING NAME FOR THE HOLE. ',//,
     C  5X,'NAMES MUST BEGIN WITH A LETTER',//,
     D  5X,'NAMES MAY NOT CONTAIN PERIODS(.)')
      S T O P
30    WRITE(LP,31)
31    FORMAT(//,5X,'E R R O R  3',//,
     B  5X,'EXPECTING NUMBER OR NUM, BUT NOT FOUND')
      S T O P
40    WRITE(LP,41)
41    FORMAT(//,5X,'E R R O R  4',//,
     C  5X,'EXPECTING OF, BUT NOT FOUND')
      S T O P
50    WRITE(LP,51)
51    FORMAT(//,5X,'E R R O R  5',//,
     D  5X,'EXPECTING BORINGS OR BOR, BUT NOT FOUND')
      S T O P
60    WRITE(LP,61)
61    FORMAT(//,5X,'E R R O R  6',//,
     E  5X,'EXPECTING AN INTEGER QUANTITY FOR THE NUMBER OF BORINGS')
      S T O P
70    WRITE(LP,71)
71    FORMAT(//,5X,'E R R O R  7',//,
     G  5X,'EXPECTING UNIT DESIGNATION, IMPERIAL(IMP) OR METRIC(MET)')
      S T O P
80    WRITE(LP,81)
81    FORMAT(//,5X,'E R R O R  8',//,
```

```
     H  5X,'EXPECTING STRATA OR STR, BUT NOT FOUND')
      S T O P
90    WRITE(LP,91)
91    FORMAT(//,5X,'E R R O R  9',//,
     I  5X,'EXPECTING ELEVATION OR E, BUT NOT FOUND')
      S T O P
100   WRITE(LP,101)
101   FORMAT(//,5X,'E R R O R  10',//,
     J  5X,'EXPECTING REAL VALUE FOR ELEVATION, BUT NOT FOUND')
      S T O P
110   WRITE(LP,111)
111   FORMAT(//,5X,'E R R O R  11',//,
     K  5X,'EXPECTING LATITUDE OR LAT, BUT NOT FOUND')
      S T O P
120   WRITE(LP,121)
121   FORMAT(//,5X,'E R R O R  12',//,
     L  5X,'EXPECTING A REAL VALUE FOR LATITUDE')
      S T O P
130   WRITE(LP,131)
131   FORMAT(//,5X,'E R R O R  13',//,
     M  5X,'EXPECTING LONGITUDE OR LON, BUT NOT FOUND')
      S T O P
140   WRITE(LP,141)
141   FORMAT(//,5X,'E R R O R  14',//,
     N  5X,'EXPECTING A REAL VALUE FOR LONGITUDE')
      S T O P
150   WRITE(LP,151)
151   FORMAT(//,5X,'E R R O R  15',//,
     O  5X,'EXPECTING DEPTH OR D, BUT NOT FOUND')
      S T O P
160   WRITE(LP,161)
161   FORMAT(//,5X,'E R R O R  16',//,
     P  5X,'EXPECTING A REAL NUMBER VALUE FOR DEPTH')
      S T O P
170   WRITE(LP,171)
171   FORMAT(//,5X,'E R R O R  17',//,
     Q  5X,'EXPECTING AN INTEGER VALUE FOR THE NUMBER OF STRATA')
      S T O P
180   WRITE(LP,181)
181   FORMAT(//,5X,'E R R O R  18',//,
     R  5X,'EXPECTING THE NAME FOR THE STRATA',//,
     S  5X,'NAMES MAY NOT CONTAIN PERIODS (.)')
      S T O P
190   WRITE(LP,191)
191   FORMAT(//,5X,'E R R O R  19',//,
     T  5X,'EXPECTING TO FIND SCALE OR SC, BUT NOT FOUND')
      S T O P
200   WRITE(LP,201)
201   FORMAT(//,5X,'E R R O R  20',//,
     U  5X,'************   AVAILABLE MESSAGE AREA   ******')
      S T O P
210   WRITE(LP,211)
211   FORMAT(//,5X,'E R R O R  21',//,
     V  5X,'************   AVAILABLE MESSAGE AREA   ***')
```

```
      S T O P
220   WRITE(LP,221)
221   FORMAT(//,5X,'E R R O R  22',//,
     W  5X,'EXPECTING SECTIONS OR SEC, BUT NOT FOUND')
      S T O P
230   WRITE(LP,231)
231   FORMAT(//,5X,'E R R O R  23',//,
     X  5X,'EXPECTING AN INTEGER VALUE FOR THE NUMBER OF SECTIONS')
      S T O P
240   WRITE(LP,241)
241   FORMAT(//,5X,'E R R O R  24',//,
     Y  5X,'EXPECTING IN, BUT NOT FOUND')
      S T O P
250   WRITE(LP,251)
251   FORMAT(//,5X,'E R R O R  25',//,
     Z  5X,'EXPECTING AN INTEGER VALUE FOR THE NUMBER',//,
     1  5X,'OF BORINGS IN THE SECTION')
      S T O P
260   WRITE(LP,261)
261   FORMAT(//,5X,'E R R O R  26',//,
     2  5X,'EXPECTING ARE, BUT NOT FOUND')
      S T O P
270   WRITE(LP,271)
271   FORMAT(//,5X,'E R R O R  27',//,
     3  5X,'EXPECTING AN INTEGER VALUE FOR THE BORINGS SEQUENTIAL NAME')
      S T O P
280   WRITE(LP,281)
281   FORMAT(//,5X,'E R R O R   28',//,
     4  5X,'***********  AVAILABLE MESSAGE AREA  ****')
      S T O P
290   WRITE(LP,291)
291   FORMAT(//,5X,'E R R O R  29',//,
     5  5X,'EXPECTING FIELDS OR GLOBAL, BUT NOT FOUND.')
      S T O P
300   WRITE(LP,301)
301   FORMAT(//,5X,'E R R O R  30',//,
     6  5X,'EXPECTING VIEWING OR V, BUT NOT FOUND')
      S T O P
310   WRITE(LP,311)
311   FORMAT(//,5X,'E R R O R  31',//,
     7  5X,'EXPECTING XSECTION OR XSEC, BUT NOT FOUND')
      S T O P
320   WRITE(LP,321)
321   FORMAT(//,5X,'E R R O R  32',//,
     8  5X,'EXPECTING A REAL VALUE FOR THE VIEWING ELEV., IN DEGREES')
      S T O P
330   WRITE(LP,331)
331   FORMAT(//,5X,'E R R O R  33',//,
     9  5X,'*************** AVALIABLE MESSAGE AREA ****************')
      S T O P
340   WRITE(LP,341)
341   FORMAT(//,5X,'E R R O R  34',//,
     A  5X,'EXPECTING ANGLE OR A, BUT NOT FOUND')
      S T O P
```

```
350   WRITE(LP,351)
351   FORMAT(//,5X,'E R R O R  35',//,
     B  5X,'EXPECTING REAL VALUE FOR THE ANGLE OF VIEWING, IN DEGREES')
      S T O P
360   WRITE(LP,361)
361   FORMAT(//,5X,'E R R O R  36',//,
     C  5X,'EXPECTING FINAL OR FIN, BUT NOT FOUND')
      S T O P
370   WRITE(LP,371)
371   FORMAT(//,5X,'E R R O R  37',//,
     D  5X,'EXPECTING ENGAGE OR ENG, BUT NOT FOUND.')
      S T O P
      E N D
C ***************************************************************
C *                                                             *
C *   FTOMGL                                                    *
C *                                                             *
C ***************************************************************
      SUBROUTINE FTOMGL
C
C FTOMGL CONVERTS DATA FROM IMPERIAL TO METRIC UNITS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      INTEGER * 4 HOLE,ELEV,LAT,LON,DEPTH,NU,O,STRATA,NOT
      DATA HOLE/'H'/,ELEV/'E'/,LAT/'LAT'/,LON/'LON'/,DEPTH/'D'/,
     A  NU/'NU'/,O/'O'/,STRATA/'STR'/,NOT/'NOT'/
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEXT,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE,NEXT
      IFLIP1 = 1
      DO 70 J=1,NOBOR
C
C READ IN NAME,COORDINATES AND ELEVATIONS OF BORINGS
C
       C A L L   R E A D S C
             IF (MATCH(HOLE,1)) GO TO 100
       CALL ERRGL(1)
 100         IF (LABEL(1)) GO TO 200
       CALL ERRGL(2)
 200   CALL ENTIT(NAME1(J),4,1)
             IF (MATCH(ELEV,1)) GO TO 400
       CALL ERRGL(9)
 400         IF (NUMR(ZGIVEN(J))) GO TO 500
       CALL ERRGL(10)
```

```
 500   ZKEEP(J) = ZGIVEN(J)
             IF (MATCH(LAT,3)) GO TO 600
       CALL ERRGL(11)
 600         IF (NUMR(XCOORD(J))) GO TO 700
       CALL ERRGL(12)
 700         IF (MATCH(LON,3)) GO TO 800
       CALL ERRGL(13)
 800         IF (NUMR(YCOORD(J))) GO TO 900
       CALL ERRGL(14)
 900         IF (MATCH(DEPTH,1)) GO TO 1000
       CALL ERRGL(15)
 1000        IF (NUMR(TOTDEP(J))) GO TO 1100
       CALL ERRGL(16)
 1100  TOTDEM(J)=TOTDEP(J)/3.28083
C
C CHANGE THE UNITS FROM IMPERIAL TO METRIC
C
       XCOORM(J)=XCOORD(J)/3280.83
       YCOORM(J)=YCOORD(J)/3280.83
       ZGIVEM(J)=ZGIVEN(J)/3.28083
 70    C O N T I N U E
C
C READ IN THE NUMBER OF STRATA
C
      C A L L   R E A D S C
             IF (MATCH(NU,2)) GO TO 1200
       CALL ERRGL(3)
1200         IF (MATCH(O,1)) GO TO 1300
       CALL ERRGL(4)
1300        IF (MATCH(STRATA,3)) GO TO 1400
       CALL ERRGL(8)
1400        IF (NUMI(NOSTRA)) GO TO 1500
       CALL ERRGL(17)
C
C READ IN THE STRATA NAME
C
 1500 DO 1040 K=1,NOSTRA
      C A L L   R E A D S C
             IF (LABEL(1)) GO TO 1600
       CALL ERRGL(18)
1600  CALL ENTIT(KTYPE(K),4,1)
C
C READ IN THE NAME AND DEPTH AT WHICH THE PARTICULAR SOIL
C TYPE WERE FOUND
C
       DO 1040 N=1,NOBOR
       C A L L   R E A D S C
             IF (LABEL(1)) GO TO 1700
       CALL ERRGL(2)
1700   CALL ENTIT(NAME1(N),4,1)
             IF (NUMR(STELEV(K,N))) GO TO 1800
             IF (MATCH(NOT,3)) STELEV(K,N) = -10.0
1800   STELEM(K,N)=STELEV(K,N)/3.28083
1040   C O N T I N U E
```

```
      R E T U R N
      E N D
C ***************************************************************
C *                                                             *
C *   GRID                                                      *
C *                                                             *
C ***************************************************************
      SUBROUTINE GRID
C
C GRID WILL PRODUCE THE CONTOUR MATRIX FOR THE SOIL STRATA
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      DIMENSION NODEX(25),NODEY(25),NODEZ(25,25),NODEQ(25)
      NXX = 99
      NNN = 1
            DO 20 I=1,NOSTRA
            XMIN=1.0E10
            XMAX=0.0
            YMIN=1.0E10
            YMAX=0.0
                  DO 30 J=1,NOBOR
                  IF (STELEV(I,J).LT.0.0) GO TO 30
                  IF (XCOORD(J).LT.XMIN) XMIN=XCOORD(J)
                  IF (XCOORD(J).GT.XMAX) XMAX=XCOORD(J)
                  IF (YCOORD(J).LT.YMIN) YMIN=YCOORD(J)
                  IF (YCOORD(J).GT.YMAX) YMAX=YCOORD(J)
                  ZGIVEN(J)=ZKEEP(J)-STELEV(I,J)
                  IF (I.EQ.1) ZGIVEN(J)=ZKEEP(J)
 30               C O N T I N U E
            Y = 0.0
            DO 10 LONG=1,25
            IF (LONG.EQ.1) LONGST=YMIN
            IF (LONG.EQ.2) LONGST=(YMAX-YMIN)/24.0
            Y=Y+LONGST
            NODEY(LONG)=Y
            X = 0.0
                        DO 10 LAT=1,25
                        NODEQ(LAT)=LAT
                        IF (LAT.EQ.1) LATSTE=XMIN
                        IF (LAT.EQ.2) LATSTE=(XMAX-XMIN)/24.0
                        X=X+LATSTE
                        NODEX(LAT)=X
                        CALL ZEE(X,Y,ZFIND)
                        NODEZ(LONG,LAT)=ZFIND
 10                     C O N T I N U E
```

```
            KROSS = 25
            IF (I.EQ.NOSTRA) KTYPE(I) = KTYPE(10)
            WRITE(LP,40) KTYPE(I),(NODEQ(L),L=1,25)
 60         C O N T I N U E
            WRITE(LP,50) KROSS,(NODEZ(KROSS,M),M=1,25)
            KROSS = KROSS - 1
            IF (KROSS.NE.0) GO TO 60
            WRITE(LP,70) KTYPE(I),(NODEQ(N),NODEX(N),NODEQ(N),NODEY(N),
     A      N=1,25)
            NNN = NNN + 1
 20         C O N T I N U E
 40    FORMAT(1H1,///,50X,20('*') ,/,50X,'*',18X,'*',/,50X,'*',
     A 4X,'CONTOUR GRID',2X,'*',/50X,'*',8X,'FOR',7X,'*',/,
     B 50X,'*',4X,A4,' STRATA',3X,'*',/,50X,'*',18X,'*',/,50X,
     C 20('*'),///,8X,25('ND', I2,1X),/,6X,125('-'),/ )
 50    FORMAT(1X,'PT',I2,1X,'|',25(1X,I4),/,6X,'|')
 70    FORMAT(1H1,///,51X,'COORDINATE LEGEND',/,58X,'FOR',/,
     A 54X,A4,' STRATUM',///,42X,'LATITUDE',22X,'LONGITUDE',//,
     B 25(36X,'NODE',2X,I2,2X,I10,9X,'POINT',2X,
     C I2,2X,I10,//))
      NXX = 69
      R E T U R N
      E N D
C ***************************************************************
C *                                                             *
C *   INPUTG                                                    *
C *                                                             *
C ***************************************************************
      SUBROUTINE INPUTG
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEXT,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      INTEGER * 4 NUMBER,OF,BORING,METRIC,IMPER,XSEC
     A  IN,ARE,SEC,VIEW,EL,ANGLE,FIN,STRATA
      DATA NUMBER/'NUM'/,OF/'OF'/,BORING/'BOR'/,METRIC/'MET'/,
     A SEC/'SEC'/,VIEW/'V'/,EL/'E'/,ANGLE/'A'/,FIN/'FIN'/,IMPER/'IMP'/,
     B STRATA/'STR'/,IN/'IN'/,ARE/'ARE'/,XSEC/'XSEC'/
C
C THE FOLOWING LOGICAL SUBROUTINES HAVE DUMMY ARGUMENTS:
C    --  ENDCRD, LABEL, NAME, NOSCAN, STRING, TRUE  --
C
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      IFLIP1 = 0
 30   C A L L   R E A D S C
```

```
            IF (MATCH(NUMBER,3)) GO TO 20
      CALL ERRGL(3)
20          IF (MATCH(OF,2)) GO TO 70
      CALL ERRGL(4)
70          IF (MATCH(BORING,3)) GO TO 200
      CALL ERRGL(5)
200         IF (NUMI(NOBOR)) GO TO 290
      CALL ERRGL(6)
C
C READ IN TYPE OF UNITS FEET OR METERS?
C
290         NOPNTS = NOBOR
      C A L L   R E A D S C
            IF (MATCH(IMPER,3)) CALL FTOMGL
            IF (IFLIP1.GT.0) GO TO 500
            IF (MATCH(METRIC,3)) CALL MTOFGL
            IF (IFLIP1.LT.0) GO TO 500
      CALL ERRGL(7)
500   C A L L   R E A D S C
            IF (MATCH(NUMBER,3)) GO TO 600
      CALL ERRGL(3)
600         IF (MATCH(OF,2)) GO TO 700
       CALL ERRGL(4)
700         IF (MATCH(SEC,3)) GO TO 800
       CALL ERRGL(22)
800         IF (NUMI(NOXSEC))  GO TO 900
       CALL ERRGL(23)
900    DO 950 NX = 1,NOXSEC
       C A L L   R E A D S C
            IF (MATCH(XSEC,4)) GO TO 910
       CALL ERRGL(31)
910         IF (LABEL(1)) GO TO 920
      CALL ERRGL(32)
920   CALL ENTIT(NMXSEC(NX),4,1)
            IF (MATCH(NUMBER,3)) GO TO 1000
      CALL ERRGL(3)
1000        IF (MATCH(OF,2)) GO TO 1100
      CALL ERRGL(4)
1100        IF (MATCH(BORING,3)) GO TO 1200
      CALL ERRGL(5)
1200        IF (MATCH(IN,2)) GO TO 1300
      CALL ERRGL(24)
1300        IF (MATCH(SEC,3)) GO TO 1400
      CALL ERRGL(22)
1400        IF (NUMI(NOBOSE)) GO TO 1500
      CALL ERRGL(25)
1500        IF (MATCH(BORING,3)) GO TO 1600
      CALL ERRGL(5)
1600        IF (MATCH(ARE,3)) GO TO 1700
      CALL ERRGL(26)
1700      NBXSEC(NX) = NOBOSE
      DO 1750 MX = 1,NOBOSE
            IF (NUMI(NMGEPS(NX,MX ))) GO TO 1750
      CALL ERRGL(27)
```

```
1750  C O N T I N U E
 950  C O N T I N U E
      C A L L   R E A D S C
            IF (MATCH(VIEW,1)) GO TO 2000
      CALL ERRGL(30)
2000        IF (MATCH(EL,1)) GO TO 2100
      CALL ERRGL(9)
2100        IF (NUMR(ELEVA)) GO TO 2200
      CALL ERRGL(32)
2200  C A L L   R E A D S C
            IF (MATCH(VIEW,1)) GO TO 2300
      CALL ERRGL(30)
2300        IF (MATCH(ANGLE,1)) GO TO 2400
      CALL ERRGL(34)
2400        IF (NUMR(ZANG)) GO TO 2500
      CALL ERRGL(35)
2500  C A L L   R E A D S C
            IF (MATCH(FIN,3)) GO TO 3600
      CALL ERRGL(36)
3600        IF (MATCH(STRATA,3)) GO TO 3700
      CALL ERRGL(3)
3700        IF (LABEL(1)) GO TO 3800
      CALL ERRGL(18)
3800  CALL ENTIT(NALAST,4,1)
      WRITE(LP,3801)
3801  FORMAT(////,5X,'. . . . END OF INPUT EXECUTION + + +')
      R E T U R N
      E N D
C ***************************************************************************
C *                                                                         *
C *   MTOFGL                                                                *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE MTOFGL
C
C MTOFGL CONVERTS DATA FROM METRIC TO IMPERIAL UNITS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      INTEGER * 4 HOLE,ELEV,LAT,LON,DEPTH,NU,O,STRATA,NOT
      DATA HOLE/'H'/,ELEV/'E'/,LAT/'LAT'/,LON/'LON'/,DEPTH/'D'/,
     A  NU/'NU'/,O/'O'/,STRATA/'STR'/,NOT/'NOT'/
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEXT,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL NEXT
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
```

```
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE
      IFLIP1 = -1
      DO 70 J=1,NOBOR
C
C READ IN NAME, COORDINATES AND ELEVATIONS OF BORINGS
C
       C A L L   R E A D S C
             IF (MATCH(HOLE,1)) GO TO 100
        CALL ERRGL(1)
 100          IF (LABEL(NAME1(J))) GO TO 300
       CALL ERRGL(2)
 300          IF (MATCH(ELEV,1)) GO TO 400
       CALL ERRGL(9)
 400          IF (NUMR(ZGIVEM(J))) GO TO 500
       CALL ERRGL(10)
 500          IF (MATCH(LAT,3)) GO TO 600
       CALL ERRGL(11)
 600          IF (NUMR(XCOORM(J))) GO TO 700
       CALL ERRGL(12)
 700          IF (MATCH(LON,3)) GO TO 800
       CALL ERRGL(13)
 800          IF (NUMR(YCOORM(J))) GO TO 900
       CALL ERRGL(14)
 900          IF (MATCH(DEPTH,1)) GO TO 1000
       CALL ERRGL(15)
 1000         IF (NUMR(TOTDEM(J))) GO TO 1100
       CALL ERRGL(16)
C
C CONVERT FROM METRIC TO IMPERIAL UNITS
C
 1100 XCOORD(J)=XCOORM(J)*3280.83
      YCOORD(J)=YCOORM(J)*3280.83
      ZGIVEN(J)=ZGIVEM(J)*3.28083
      TOTDEP(J)=TOTDEM(J)*3.28083
 70   C O N T I N U E
C
C READ IN THE NUMBER OF STRATA
C
      C A L L   R E A D S C
            IF (MATCH(NU,2)) GO TO 1200
      CALL ERRGL(3)
1200        IF (MATCH(O,1)) GO TO 1300
      CALL ERRGL(4)
1300      IF (MATCH(STRATA,3)) GO TO 1400
      CALL ERRGL(8)
1400      IF (NUMI(NOSTRA)) GO TO 1500
      CALL ERRGL(17)
C
C READ IN THE STRATA NAME
C
 1500       DO 1040 K=1,NOSTRA
            C A L L   R E A D S C
            IF (LABEL(KTYPE(K))) GO TO 1600
            CALL ERRGL(18)
```

```
C
C READ IN THE NAME AND DEPTH AT WHICH THE PARTICULAR SOIL
C TYPE  WERE FOUND
C
 1600             DO 1040 N=1,NOBOR
                  IF (LABEL(NAME1(N))) GO TO 1700
                  CALL ERRGL(2)
 1700             IF (NUMR(STELEM(K,N))) GO TO 1800
                  IF (MATCH(NOT,3)) STELEM(K,N) = -10.0
 1800             STELEV(K,N)=STELEM(K,N)/3.28083
 1040             C O N T I N U E
      R E T U R N
      E N D
C ****************************************************************
C *                                                              *
C *   THREED                                                     *
C *                                                              *
C ****************************************************************
      SUBROUTINE THREED(D55,IX1,IX2,IX3,IX4,IX5,IX6)
C
C THREED DRAWS THE SOIL INTERFACE ISOMETRIC VIEW
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      NOSTM1=NOSTRA-1
            DO 10 I3=1,NOSTM1
            ZEMIN2=1.0E10
            XMIN=1.0E10
            XMAX=0.0
            YMIN=1.0E10
            YMAX=0.0
            ZMIN=1.0E10
            ZMAX=0.0
                  DO 20 I4=1,NOBOR
C
C REDEFINE THE AREA ACCORDING TO THE PERTINENT BORINGS
C
                  IF (STELEV(I3,I4).LT.0.0) GO TO 20
                  IF (XCOORD(I4).LT.XMIN) XMIN=XCOORD(I4)
                  IF (XCOORD(I4).GT.XMAX) XMAX=XCOORD(I4)
                  IF (YCOORD(I4).LT.YMIN) YMIN=YCOORD(I4)
                  IF (YCOORD(I4).GT.YMAX) YMAX=YCOORD(I4)
                  ZGIVE(I4) = (ZKEEP(I4)-STELEV(I3,I4))
                  IF (ZGIVE(I4).LT.ZEMIN2) ZEMIN2=ZGIVE(I4)
 20               C O N T I N U E
C
```

```
C ADJUST THE ELEVATIONS SO THAT THEY WILL BE POSITIVE
C
            CHANGE=ABS(ZEEMIN-ZEMIN2)
            D11=XMIN
            D22=XMAX
            D33=YMIN
            D44=YMAX
            E11=ELEVA
            E22=ZANG
                  DO 30 MM=1,NOBOR
                  ZGIVEN(MM)=ZGIVE(MM)+CHANGE
                  IF (ZGIVEN(MM).LT.ZMIN) ZMIN=ZGIVEN(MM)
                  IF (ZGIVEN(MM).GT.ZMAX) ZMAX=ZGIVEN(MM)
 30               C O N T I N U E
            FACTO=(XMAX-XMIN)/(ZMAX-ZMIN)
            NNN=NNN+1
      CALL PZOSU(D11,D22,D33,D44,D55,IX1,IX2,IX3,IX4,IX5,IX6,E11,E22)
 10         C O N T I N U E
      R E T U R N
      E N D
C ********************************************************
C *                                                      *
C *   XSEC                                               *
C *                                                      *
C ********************************************************
      SUBROUTINE XSEC
C
C XSEC DETERMINES THE GEOTECHNICAL SECTIONS DEFINED BY THE USER
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      DIMENSION THICKF(10,50),THICKM(10,50),ZPREVF(50),ZPREVM(50),
     1 THELEV(10,50),THELEM(10,50),STLINM(7),ELEVST(10,50),ADDER(10,7),
     2 XDISTF(7),XDISTM(7),YDISTF(7),YDISTM(7),STLINF(7),BOTEL(50),
     3 XBO(8),TOPBO(8)
      WRITE(LP,10)
      WRITE(LP,60)
      C A L L   P L O T   (10.0,-7.0,-3)
            DO 80 I3=1,NOBOR
            NOSOIL=0
                  DO 90 I4=1,NOSTRA
                  IF (STELEV(I4,I3).GT.0.0)NOSOIL=NOSOIL+1
 90               C O N T I N U E
C
C WRITE OUT THE NAME, TOTAL DEPTH AND NUMBER OF STRATA
C ENCOUNTERED FOR EACH BORING
```

```
C
            WRITE(LP,70) NAME1(I3),I3,TOTDEP(I3),TOTDEM(I3),NOSOIL
 80         C O N T I N U E
            DO 100 I1=1,NOSTRA
C
C WRITE OUT THE STRATA NAME
C
            WRITE(LP,20) KTYPE(I1)
            WRITE(LP,50)
C
C CALCULATE THE THICKNESS OF THE RESPECTIVE STRATA
C
                  DO 100 I2=1,NOBOR
                  IF (STELEV(I1,I2).LT.0.0)GO TO 100
                  IF (I1.EQ.NOSTRA) ZPREVF(I2)=0.0
                  IF (I1.EQ.NOSTRA) ZPREVM(I2)=0.0
                  IF (I1.GT.1.AND.STELEV(I1-1,I2).GT.0.0)
     1            ZPREVF(I2)=STELEV(I1-1,I2)
                  IF (I1.GT.1.AND.STELEM(I1-1,I2).GT.0.0)
     1            ZPREVM(I2)=STELEM(I1-1,I2)
                  IF (I1.EQ.1) ZPREVF(I2)=0.0
                  IF (I1.EQ.1) ZPREVM(I2)=0.0
                  IF (I1.EQ.NOSTRA) ZPREVF(I2) = 0.0
                  IF (I1.EQ.NOSTRA) ZPREVM(I2) = 0.0
                  THICKF(I1,I2)=STELEV(I1,I2)-ZPREVF(I2)
                  THICKM(I1,I2)=STELEM(I1,I2)-ZPREVM(I2)
                  THELEM(I1,I2)=ZGIVEM(I2)-STELEM(I1,I2)
                  THELEV(I1,I2)=ZGIVEN(I2)-STELEV(I1,I2)
                  KPTNME(I1,I2) = I2
                  KEPTBO(I1)=KEPTBO(I1)+1
C
C WRITE OUT THE NAME, STRATA DEPTH AND STRATA ELEVATION
C
                  WRITE(LP,40)NAME1(I2),STELEV(I1,I2),STELEM(I1,I2),
     1 THICKF(I1,I2),THICKM(I1,I2),THELEV(I1,I2),THELEM(I1,I2)
 100              C O N T I N U E
            DO 120 N1=1,NOXSEC
            WRITE(LP,110) NMXSEC(N1)
            WRITE(LP,160)
            KOUNTX=NBXSEC(N1)
                  DO 130 N2=1,KOUNTX
                  N3=NMGEPS(N1,N2)
                  N4=NMGEPS(N1,N2+1)
                  WRITE(LP,140) NMGEPS(N1,N2)
                  IF (N2.EQ.NBXSEC(N1))GO TO 130
                  IF (N2.EQ.1)ADDER(N1,1)=0.0
                  YDISTM(N2)=YCOORM(N4)-YCOORM(N3)
                  YDISTF(N2)=YCOORD(N4)-YCOORD(N3)
                  XDISTM(N2)=XCOORM(N4)-XCOORM(N3)
                  XDISTF(N2)=XCOORD(N4)-XCOORD(N3)
         STLINF(N2)=SQRT(XDISTF(N2)*XDISTF(N2)+YDISTF(N2)*YDISTF(N2))
         STLINM(N2)=SQRT(XDISTM(N2)*XDISTM(N2)+YDISTM(N2)*YDISTM(N2))
                  ADDER(N1,N2+1)=ADDER(N1,N2)+STLINF(N2)
C
```

```
C WRITE OUT THE CROSS SECTION DATA, WITH THE DISTANCES
C BETWEEN THE BORINGS
C
      WRITE(LP,150) XDISTF(N2),XDISTM(N2),YDISTF(N2),YDISTM(N2),
     1 STLINF(N2),STLINM(N2)
 130              C O N T I N U E
 120         C O N T I N U E
             DO 210 M1=1,NOXSEC
             ELMAX=.1E-10
             ELMIN=.1E10
C
C PLOT THE CROSS SECTIONS
C
             C A L L   N E W P E N (1)
             C A L L   P L O T   (8.0,2.0,-3)
             CALL SYMBOL(0.,-1.0,0.14,'CROSS SECTION -',0.0,15)
             CALL SYMBOL(2.25,-1.,0.14,NMXSEC(M1),0.0,4)
             KOUNT2=NBXSEC(M1)
                   DO 200 M2=1,KOUNT2
                   M3=NMGEPS(M1,M2)
                   BOTEL(M3)=ZGIVEN(M3)-TOTDEP(M3)
                   IF (BOTEL(M3).LT.ELMIN) ELMIN=BOTEL(M3)
                   IF (ZGIVEN(M3).GT.ELMAX) ELMAX=ZGIVEN(M3)
 200               C O N T I N U E
             ELMIN=ELMIN+1000.0
             ELMAX=ELMAX+1000.0
             ITRUNC=ELMAX/10.0
             LINE=ITRUNC*10
             NUMEL=LINE-990
             SCALEY=(ELMAX-ELMIN)/5.0
             SCALEX=SCALEY*4.0
             DEC=0.0
                   DO 400 N5=1,10
                   DEC=DEC+10.0/SCALEY
                   IF (N5.EQ.1) DEC=0.0
                   YLINE=((LINE-ELMIN)/SCALEY)-DEC
                   IF (YLINE.LT.0.0) GO TO 500
                   XEND=ADDER(M1,KOUNT2)/SCALEX
                   NUMEL=NUMEL-10
                   TUMEL=NUMEL
C
C DRAW THE TEN FOOT INTERVAL LINES
C
                   CALL NUMBER(-0.6,YLINE,0.14,TUMEL,0.0,0)
                   C A L L   P L O T   (0.0,YLINE,3)
                   C A L L   P L O T   (XEND,YLINE,2)
 400               C O N T I N U E
 500         C O N T I N U E
                   DO 300 K1=1,KOUNT2
                   K3=NMGEPS(M1,K1)
                   TOPBOR=((ZGIVEN(K3)+1000.0)-ELMIN)/SCALEY
                   BOTBOR=((BOTEL(K3)+1000.0)-ELMIN)/SCALEY
                   XBOR=ADDER(M1,K1)/SCALEX
                   IF (K1.EQ.1) XBOR=0.0
```

```
                XBO(K1)=XBOR
                TOPBO(K1)=TOPBOR
                ONTOP=TOPBOR+0.25
C
C DRAW THE BORINGS
C
                C A L L   N E W P E N (3)
                CALL SYMBOL(XBOR,ONTOP,0.14,NAME1(K3),0.0,4)
                C A L L   P L O T   (XBOR,TOPBOR,3)
                C A L L   P L O T   (XBOR,BOTBOR,2)
 300            C O N T I N U E
          C A L L   N E W P E N (2)
C
C DRAW THE GROUND SURFACE OF THE CROSS SECTION
C
                DO 800 LNETOP=1,KOUNT2
                IF (LNETOP.EQ.1)C A L L  P L O T(0.0,TOPBO(LNETOP),3)
                C A L L   P L O T   (XBO(LNETOP),TOPBO(LNETOP),2)
 800            C O N T I N U E
                NOSTM1=NOSTRA-1
                DO 600 ML=1,NOSTM1
                DO 600 IL=1,KOUNT2
                IPEN=2
                K3=NMGEPS(M1,IL)
                XBORR=ADDER(M1,IL)/SCALEX
                ELEVST(ML,K3)=THELEV(ML,K3)+1000.0
                YSTRAT=(ELEVST(ML,K3)-ELMIN)/SCALEY
                IF (IL.EQ.1) GO TO 700
                IF (STELEV(ML,K3PREV).LT.0.0) IPEN=3
                IF (STELEV(ML,K3).LT.0.0) IPEN=3
C
C DRAW THE STRATA INTERFACE LINES
C
                C A L L   P L O T   (XBORR,YSTRAT,IPEN)
                GO TO 600
 700            STSTRA=(ELEVST(ML,K3)-ELMIN)/SCALEY
                C A L L   P L O T   (0.0,STSTRA,3)
                K3PREV=K3
 600            C O N T I N U E
          C A L L   P L O T   (XEND,-7.0,-3)
 210      C O N T I N U E
     C A L L   P L O T   (XEND,-7.0,-3)
 10   FORMAT(1H1,/,50X,30('*'),/,2(50X,'*',28X,'*',/),50X,'*',7X,
     1 'CROSS SECTIONS',7X,'*',/,2(50X,'*',28X,'*',/),50X,30('*'),/)
 20   FORMAT(1H1,////,52X,'STRATA DESIGNATION',//,54X,'---- ',A4,' ----
     1 ',/////)
 50   FORMAT(18X,'BORING',19X,'DEPTH TO',22X,'THICKNESS',22X,'ELEVATION'
     1 ,/,39X,'BOTTOM OF STRATA',18X,'OF STRATA',22X,'OF STRATA',/,
     2 42X,'FEET/MTRS',22X,'FEET/MTRS',22X,'FEET/MTRS',////)
 40   FORMAT(19X,A4,3(16X,F7.3,'/',F7.3)/)
 110  FORMAT(1H1,////,52X,'CROSS SECTION',//,48X,'---- NUMBER ',
     1 A4,' ----',//)
 60   FORMAT(////,32X,'BORING',8X,'GEPS',18X,'TOTAL DEPTH',11X,
     1 'NUMBER OF SOILS',/,46X,'NAME',42X,'ENCOUNTERED',/,
```

```
     2 69X,'FEET/MTRS',//)
 70   FORMAT(33X, A4,10X,I2,17X, F7.3,'/',F7.3,16X,I1,/)
 140  FORMAT(20X,I2,/)
 150  FORMAT(35X,  F7.2,'/',F6.3,11X,F7.2,'/',F6.3,13X,F7.2,'/',F6.3,/)
 160  FORMAT(20X,'GEPS',15X,'LATITUDE',17X,'LONGITUDE',15X,'STRAIGHT LIN
     1E',  /,20X,'NAME',15X,'DISTANCE',17X,'DISTANCE',19X,'DISTANCE',//,
     2 40X,'FT/KM',20X,'FT/KM',22X,'FT/KM',//)
      R E T U R N
      E N D
      SUBROUTINE FIELDS
C ***************************************************************
C *                                                             *
C *    F I E M O D                                              *
C *                                                             *
C ***************************************************************
C  **
C  **
C  **
C  **
C  **
C  **
C  **
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEST,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      INTEGER * 4 START,END2,COM,STAT,SEPE
      DATA START/'STAR'/,END2/'END'/,COM/'COM'/,NOG/'NOG'/,
     A NOD/'NOD'/,NOS/'NOS'/,NOM/'NOM'/,STAT/'STAT'/,
     B SEPE/'SEP'/,LINE/'LINE'/
5     C A L L   R E A D S C
            IF (MATCH(START,4)) GO TO 10
            IF (MATCH(END2,3)) S T O P
      CALL ERRFI(8)
10    C A L L   R E A D S C
             IF (MATCH(COM,3)) GO TO 15
             IF (MATCH(STAT,4)) GO TO 100
      CALL ERRFI(11)
15           IF (MATCH(NOG,3)) GO TO 25
          IX2 = 9
             GO TO 30
25        IX2 = -9
30           IF (MATCH(NOD,3)) GO TO 35
          IX3 = 9
             GO TO 40
35        IX3 = -9
40           IF (MATCH(NOS,3)) GO TO 45
          IX4 = 9
```

```
            GO TO 50
45        IX4 = -9
            IF (MATCH(NOM,3)) GO TO 55
50        IX5 = 9
          IX6 = 9
            GO TO 10
55        IX5 = -9
          IX6 = 9
            GO TO 10
100         IF (MATCH(SEPE,3)) GO TO 110
130         IF (MATCH(LINE,4)) GO TO 120
      CALL ERRFI(12)
110         IF (NUMR(D55)) GO TO 130
120         IF (NUMI(IX1)) GO TO 140
C
C READ IN ALL INPUT
C
140   CALL INPUTF
C
C CALCULATE THE PERIMETER OF THE SITE
C
      E11 = ELEVA
      E22 = ZANG
      XMIN=1.0E10
      XMAX=0.0
      YMIN=1.0E10
      YMAX=0.0
      ZMAX=0.0
      ZMIN=1.0E10
      DO 31 M=1,NOPNTS
            IF (XCOORD(M).LT.XMIN) XMIN=XCOORD(M)
            IF (XCOORD(M).GT.XMAX) XMAX=XCOORD(M)
            IF (YCOORD(M).LT.YMIN) YMIN=YCOORD(M)
            IF (YCOORD(M).GT.YMAX) YMAX=YCOORD(M)
            IF (ZGIVEN(M).LT.ZMIN) ZMIN=ZGIVEN(M)
            IF (ZGIVEN(M).GT.ZMAX) ZMAX=ZGIVEN(M)
 31         C O N T I N U E
      DIFFX=XMAX-XMIN
      DIFFY=YMAX-YMIN
      DIFFZ=ZMAX-ZMIN
      DIFMX=DIFFX/3.28083
      DIFMY=DIFFY/3.28083
      XMIM=XMIN/3.28083
      YMIM=YMIN/3.28083
      YMAM=YMAX/3.28083
      XMAM=XMAX/3.28083
      MISIX=XMIN+DIFFX/2.0
      MISIY=YMIN+DIFFX/2.0
      FACTO=(XMAX-XMIN)/(ZMAX-ZMIN)
      D11=XMIN
      D22=XMAX
      D33=YMIN
      D44=YMAX
C
```

```
C OPEN THE PLOT DATA SET
C
      CALL PLOTS
C
C CALL THE BORING LOCATION PLAN SUBROUTINE
C
      CALL PLAN
C
C DRAW THE ISOMETRIC VIEW OF THE GROUND SUFACE
C
      CALL PZOSU(D11,D22,D33,D44,D55,IX1,IX2,IX3,IX4,IX5,IX6,E11,E22)
      S T O P
      E N D
C ***********************************************************************
C *                                                                     *
C *    ERRFI                                                            *
C *                                                                     *
C ***********************************************************************
      SUBROUTINE ERRFI(NUMERR)
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      WRITE(LP,1000)
1000  FORMAT(1H1,/,2(5X,25('*'),/),2(5X,2('*'),21X,2('*'),/),5X,2('*'),
     A  8X,'INPUT',8X,2('*'),/,5X,2('*'),8X,'ERROR',8X,2('*'),/,
     B  2(5X,2('*'),21X,2('*'),/),2(5X,25('*'),/))
           GO TO(10,20,30,40,50,60,70,80,90,100,110),NUMERR
      R E T U R N
10    WRITE(LP,11)
11    FORMAT(//,5X,'E R R O R  1',//,
     A  5X,'EXPECTING POINT OR P, BUT NOT FOUND')
      S T O P
20    WRITE(LP,21)
21    FORMAT(//,5X,'E R R O R  2',//,
     B  5X,'EXPECTING AN INTEGER VALUE FOR THE POINT')
      S T O P
30    WRITE(LP,31)
31    FORMAT(//,5X,'E R R O R  3',//,
     C  5X,'EXPECTING LATITUDE OR LA, BUT NOT FOUND')
      S T O P
40    WRITE(LP,41)
41    FORMAT(//,5X,'E R R O R  4',//,
     D  5X,'EXPECTING A REAL VALUE FOR LATITUDE')
      S T O P
50    WRITE(LP,51)
51    FORMAT(//,5X,'E R R O R  5',//,
     D  5X,'EXPECTING LONGITUDE OF LO, BUT NOT FOUND')
      S T O P
60    WRITE(LP,61)
61    FORMAT(//,5X,'E R R O R  6',//,
     E  5X,'EXPECTING A REAL VALUE FOR LONGITUDE')
      S T O P
70    WRITE(LP,71)
71    FORMAT(//,5X,'E R R O R  7',//,
     G  5X,'EXPECTING UNIT DESIGNATION, IMPERIAL(IMP) OR METRIC(MET)')
      S T O P
```

```
80     WRITE(LP,81)
81     FORMAT(//,5X,'E R R O R  8',//,
      H  5X,'EXPECTING START(STAR), CONTINUE(CON) OR END(END)')
       S T O P
90     WRITE(LP,91)
91     FORMAT(//,5X,'E R R O R  9',//,
      I  5X,'EXPECTING VIEWING OR VIEW, BUT NOT FOUND')
       S T O P
100    WRITE(LP,101)
101    FORMAT(//,5X,'E R R O R  10',//,
      J  5X,'EXPECTING ELEVATION OR ELEV, BUT NOT FOUND')
       S T O P
110    WRITE(LP,111)
111     FORMAT(//,5X,'E R R O R  11',//,
      K 5X,'EXPECTING COMMANDS OR STATISTICS, BUT NOT FOUND')
       S T O P
       E N D
C ***************************************************************
C *                                                             *
C *   FTOMFI                                                    *
C *                                                             *
C ***************************************************************
C
      SUBROUTINE FTOMFI
C
C FTOMFI CONVERTS DATA FROM IMPERIAL TO METRIC UNITS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      INTEGER * 4 POIN,ELEV,LAT,LON,END2
      DATA POIN/'P'/,ELEV/'E'/,LAT/'LA'/,LON/'LO'/,END2/'END'/
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEST,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      IFLIP1 = 1
C
C READ IN NAME, COORDINATES AND ELEVATIONS OF BORINGS
C
10     C A L L   R E A D S C
           IF (MATCH(END2,3)) R E T U R N
           IF (MATCH(POIN,1)) GO TO 100
       CALL ERRFI(1)
100    NOPNTS = NOPNTS + 1
       J=NOPNTS
            IF (NUMI (NAME1(J))) GO TO 500
       CALL ERRFI(2)
 500         IF (MATCH(LAT,2)) GO TO 600
       CALL ERRFI(3)
 600         IF (NUMR(XCOORD(J))) GO TO 700
```

```
      CALL ERRFI(4)
 700        IF (MATCH(LON,2)) GO TO 800
      CALL ERRFI(5)
 800        IF (NUMR(YCOORD(J))) GO TO 900
      CALL ERRFI(6)
 900        IF (MATCH(ELEV,1)) GO TO 400
      CALL ERRFI(9)
 400        IF (NUMR(ZGIVEN(J))) GO TO 1100
      CALL ERRFI(10)
C
C CHANGE THE UNITS FROM IMPERIAL TO METRIC
C
 1100    J=NOPNTS
         XCOORM(J)=XCOORD(J)/3.28083
         YCOORM(J)=YCOORD(J)/3.28083
         ZGIVEM(J)=ZGIVEN(J)/3.28083
      GO TO 10
      E N D
C ****************************************************************
C *                                                              *
C *   INPUTF                                                     *
C *                                                              *
C ****************************************************************
      SUBROUTINE INPUTF
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEST,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      DATA VIEW/'VIEW'/,EL/'ELEV'/,METRIC/'MET'/,IMPER/'IMP'/,
     A ANG/'ANG'/,AREA99/'AREA'/
C
C THE FOLOWING LOGICAL SUBROUTINES HAVE DUMMY ARGUMENTS:
C    -- ENDCRD, LABEL, NAME, NOSCAN, STRING, TRUE --
C
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      C A L L   R E A D S C
            IF (MATCH(VIEW,4)) GO TO 10
      CALL ERRFI(9)
10          IF (MATCH(EL,4)) GO TO 20
      CALL ERRFI(10)
20          IF (NUMR(ELEVA)) GO TO 30
      CALL ERRFI(11)
30          IF (MATCH(ANG,3)) GO TO 40
      CALL ERRFI(12)
40          IF (NUMR(ZANG)) GO TO 65
      CALL ERRFI(13)
65    C A L L   R E A D S C
            IF (MATCH(AREA99,4)) GO TO 70
      CALL ERRFI(16)
```

```
70          IF (LABEL(1)) GO TO 80
      CALL ERRFI(17)
80    CALL ENTIT(LTYPE,4,1)
C
C READ IN TYPE OF UNITS FEET OR METERS?
C
290    C A L L   R E A D S C
            IF (MATCH(IMPER,3)) CALL FTOMFI
            IF (IFLIP1.GT.0) GO TO 3800
            IF (MATCH(METRIC,3)) CALL MTOFFI
            IF (IFLIP1.LT.0) GO TO 3800
      CALL ERRFI(7)
3800  WRITE(LP,3801)
3801  FORMAT(////,5X,'. . . . END OF INPUT EXECUTION + + +')
      R E T U R N
      E N D
C ***************************************************************************
C *                                                                         *
C *   MTOFFI                                                                *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE MTOFFI
C
C MTOFFI CONVERTS DATA FROM METRIC TO IMPERIAL UNITS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      INTEGER * 4 POIN,ELEV,LAT,LON,END1
      DATA POIN/'P'/,ELEV/'E'/,LAT/'LA'/,LON/'LO'/,END1/'END'/
      COMMON/SCANER/ ENTITY(20),IVAL(2),MODE,NCHAR,NWD,NEST,ICOLMN
      EQUIVALENCE(VALUE,IVALUE,IVAL(1))
      LOGICAL ENDCRD, ENDFIL, INTEGR, LABEL, MATCH, NAME, NOSCAN,
     A  NUMI, NUMR, POINT, REALN, SEP, STRING, TRUE, NEXT
      IFLIP1 = -1
C
C READ IN NAME,COORDINATES AND ELEVATIONS OF BORINGS
C
10     C A L L   R E A D S C
            IF (MATCH(END1,3)) R E T U R N
            IF (MATCH(POIN,1)) GO TO 100
       CALL ERRFI(1)
100    NOPNTS = NOPNTS + 1
       J=NOPNTS
            IF (NUMI(NAME1(J))) GO TO 500
       CALL ERRFI(2)
 500         IF (MATCH(LAT,2)) GO TO 600
       CALL ERRFI(3)
 600         IF (NUMR(XCOORM(J))) GO TO 700
       CALL ERRFI(4)
 700         IF (MATCH(LON,2)) GO TO 800
```

```
      CALL ERRFI(5)
 800        IF (NUMR(YCOORM(J))) GO TO 900
      CALL ERRFI(6)
 900        IF (MATCH(ELEV,1)) GO TO 400
      CALL ERRFI(9)
 400        IF (NUMR(ZGIVEM(J))) GO TO 1100
      CALL ERRFI(10)
C
C CONVERT FROM METRIC TO IMPERIAL UNITS
C
        J=NOPNTS
 1100   XCOORD(J)=XCOORM(J)*3.28083
        YCOORD(J)=YCOORM(J)*3.28083
        ZGIVEN(J)=ZGIVEM(J)*3.28083
     GO TO 10
     E N D
C
C
C
C
C ====================================================================
C ====================================================================
C ==                                                              ==
C ==                                                            ==
C ==         C O L L E C T I V E   N E S T                    ==
C ==                                                          ==
C ==                                                            ==
C ====================================================================
C ====================================================================
C ==
C ==
C ==
C ==
C ==
C ==
C ==
C
C
C  THE COLLECTIVE NEST CONTAINS THE SUBPROGRAMS WHICH ARE
C  SHARED BY BOTH OF THE GEPS NETWOPKS.
C
C
      SUBROUTINE CHECK(SW,SW2,ISTAR,IFOR,XPOS1,XPOS2,YPOS,YNXT,MK)
C ********************************************************************
C *                                                                  *
C *  CHECK                                                           *
C *                                                                  *
C ********************************************************************
C
C
C CHECK WILL DETERMINE IF A HIDDEN LINE IS BEING DRAWN
C
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      IXX=IFOR-1
```

```
   IF (MK) 65,65,64
65 SW=-1.
   SW2=-1.
   DE1=IFOR-ISTAR
   IF (DE1.LT.1.0)  DE1=1.0
   XNC=(XPOS2-XPOS1)/DE1
   GO TO 66
64 DIFD=CKD(IFOR)-XPOS2
   DIFU=XPOS2-CK(IFOR)
   IF (DIFU) 11,11,10
11 IF (SW) 14,14,21
14 IF (DIFD) 22,22,50
22 IF (SW2) 99,99,55
10 IF (SW2) 31,31,55
31 DE1=IFOR-ISTAR
   IF (DE1.LT.1.) DE1=1.
   XNC=(XPOS2-XPOS1)/DE1
   IF (SW) 12,12,13
13 C A L L   P L O T   (XPOS2,YNXT,2)
66 XLOS=XPOS1
        DO 30 I=ISTAR,IXX
        CK(I)=XLOS
        XLOS=XLOS+XNC
30      C O N T I N U E
   R E T U R N
12 CALL LINX(XPOS1,XPOS2,YPOS,YNXT,XX,YY,KK,IFOR,ISTAR,1)
   XLOS=XX
        DO 35 I=KK,IXX
        CK(I)=XLOS
        XLOS=XLOS+XNC
35      C O N T I N U E
   C A L L   P L O T   (XX,YY,3)
   C A L L   P L O T   (XPOS2,YNXT,2)
   SW=1.
   R E T U R N
21 CALL LINX(XPOS1,XPOS2,YPOS,YNXT,XX,YY,KK,IFOR,ISTAR,1)
   DE1=IFOR-ISTAR
   IF (DE1.LT.1.) DE1=1.
   XNC=(XPOS2-XPOS1)/DE1
16 XLOS=XPOS1
         DO 40 I=ISTAR,KK
         CK(I)=XLOS
         XLOS=XLOS+XNC
40       C O N T I N U E
   C A L L   P L O T   (XX,YY,2)
   SW=-1.
   IF (DIFD) 99,99,50
50 DE1=IFOR-ISTAR
   IF (DE1.LT.1.) DE1=1.
   XNC=(XPOS2-XPOS1)/DE1
   IF (SW2) 51,51,52
52 XLOS=XPOS1
         DO 53 I=ISTAR,IXX
         CKD(I)=XLOS
```

```
            XLOS=XLOS+XNC
   53       C O N T I N U E
      C A L L   P L O T   (XPOS2,YNXT,2)
      R E T U R N
   51 SW2=1
      CALL LINX(XPOS1,XPOS2,YPOS,YNXT,XX,YY,KK,IFOR,ISTAR,2)
      XLOS=XX
            DO 54 I=KK,IXX
            CKD(I)=XLOS
            XLOS=XLOS+XNC
   54       C O N T I N U E
      C A L L   P L O T   (XX,YY,3)
      C A L L   P L O T   (XPOS2,YNXT,2)
      R E T U R N
   55 SW2=-1.
      DE1=IFOR-ISTAR
      IF (DE1.LT.1.)  DE1=1.
      XNC=(XPOS2-XPOS1)/DE1
      CALL LINX(XPOS1,XPOS2,YPOS,YNXT,XX,YY,KK,IFOR,ISTAR,2)
      XLOS=XPOS1
            DO 56 I=ISTAR,KK
            CKD(I)=XLOS
            XLOS=XLOS+XNC
   56       C O N T I N U E
      C A L L   P L O T   (XX,YY,2)
      IF (DIFU) 99,99,12
   99 R E T U R N
      E N D
      SUBROUTINE LINX(XPOS1,XPOS2,YPOS,YNXT,XX,YY,KK,IFOR,ISTAR,NN)
C ***************************************************************************
C *                                                                         *
C *  LINX                                                                   *
C *                                                                         *
C ***************************************************************************
C
C
C LINX WILL CREATE THE SURFACE LINES TO BE PLOTTED
C
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      DE1=XPOS2-XPOS1
      IF (DE1.EQ.0.) DE1=.1E+50
      C=(YNXT-YPOS)/DE1
      D=YPOS-(XPOS1*C)
      GO TO(1,2),NN
    1 DE1=CK(IFOR)-CK(ISTAR)
      GO TO 3
    2 DE1=CKD(IFOR)-CKD(ISTAR)
    3 IF (DE1.EQ.0.)  DE1=.1E+50
      A=(YNXT-YPOS)/DE1
      GO TO(4,5),NN
    4 B=YPOS-(CK(ISTAR)*A)
      GO TO 6
    5 B=YPOS-(CKD(ISTAR)*A)
    6 DE1=A-C
```

```
      IF (DEl.EQ.0.) GO TO 10
      XX=(D-B)/DEl
      YY=XX*A+B
      KK=YY*(-100.)
      R E T U R N
   10 XX=XPOS2
      YY=YNXT
      KK=IFOR
      R E T U R N
      E N D
C ***************************************************************
C *                                                             *
C *   PLAN                                                      *
C *                                                             *
C ***************************************************************
      SUBROUTINE PLAN
C
C PLAN DRAWS THE PLAN FOR THE SITE
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      WRITE(LP,1)
C
C DRAW THE PERIMETER OF THE SITE
C
      C A L L   P L O T   (6.0,-15.0,-3)
      CALL SCALE (XCOORM,7.5,NOPNTS,1)
      CALL SCALE (YCOORM,7.5,NOPNTS,1)
      J1=NOPNTS+1
      J2=NOPNTS+2
      PRIVAX=XCOORM(J1)
      DELTX=XCOORM(J2)
      PRIVAY=YCOORM(J1)
      DELTY=YCOORM(J2)
      ORIGX=(XMIM-PRIVAX)/DELTX+6.0
      ORIGY=(YMIM-PRIVAY)/DELTY+6.0
      C A L L   N E W P E N (3)
      CALL AXIS (6.0,6.0,'LATITUDE',-8,7.5,0.0,PRIVAX,DELTX)
      CALL AXIS (13.5,6.0,' ',-1,7.5  ,90.0,PRIVAY,DELTY)
      CALL AXIS (6.0,13.5,' ', 1,7.5,0.0,PRIVAX,DELTX)
      CALL AXIS (6.0, 6.0,'LONGITUDE', 9,7.5, 90.0,PRIVAY,DELTY)
      C A L L   N E W P E N (1)
      CALL PLOT (ORIGX,ORICY,-3)
C
C DRAW THE BORINGS ON THE PLAN MAP
C
            DO 110 I=1,NOPNTS
            XCOOR1=(XCOORM(I)-XMIM)/DELTX
            YCOOR1=(YCOORM(I)-YMIM)/DELTY
            YCOOR2=YCOOR1+0.10
```

```
            CALL SYMBOL (XCOOR1,YCOOR1,0.07,10,0.0,-1)
            CALL SYMBOL(XCOOR1,YCOOR2,0.10,NAME1(I),0.0,4)
 110        C O N T I N U E
C
C DRAW THE NORTH ARROW
C
      XMOVED = 11.0
      ARTIPX=XMOVED-1.5
      AMOVE=ARTIPX-0.17
      CALL SYMBOL (ARTIPX,6.5,0.56,2,0.0,-1)
      CALL PLOT (ARTIPX,4.5,2)
      CALL SYMBOL (AMOVE,5.0,0.70,85,0.0,-1)
 120  WRITE(LP,130)
      WRITE (LP,140)
            DO 4 J=1,NOPNTS
            WRITE(LP,2) NAME1(J),XCOORD(J),XCOORM(J),YCOORD(J),
     1      YCOORM(J),ZGIVEN(J),ZGIVEM(J)
 4          C O N T I N U E
      C A L L   P L O T   (6.0,-15.0,-3)
 1    FORMAT(1H1,/,50X,30('*'),/,2(50X,'*',28X,'*',/),50X,'*',4X,
     1 'BORING LOCATION PLAN',4X,'*',/,2(50X,'*',28X,'*',/),
     2 50X,30('*')////)
 2    FORMAT(27X,A4,2(6X,F9.2,'/',F9.4),8X,F9.2,'/',F9.4,/)
 130  FORMAT(28X,'BORING',9X,'LATITUDE',17X,'LONGITUDE',18X,
     1 'ELEVATION',//)
 140  FORMAT(44X,'FT/KM',20X,'FT/KM',20X,'FEET/MTRS',////)
      R E T U R N
      E N D
C ***************************************************************************
C *                                                                         *
C *  PZOSU                                                                  *
C *                                                                         *
C ***************************************************************************
C
C
C PZOSU DRAWS THE BLOCK DIAGRAMS
C
      SUBROUTINE PZOSU(D11,D22,D33,D44,D55,IX1,IX2,IX3,IX4,IX5,IX6,E11,
     * E22)
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      XMIN=D11
      XMAX=D22
      YMIN=D33
      YMAX=D44
      XRES=D55
      YRES=IX1
      IGRID=IX2
      IDROP=IX3
      IAXES=IX4
      MOVE=IX5
      IPLAN=IX6
```

```
      ELEVA=E11
      ZANG=E22
      C A L L   N E W P E N (3)
      C A L L   P L O T   (1.0,-15.0,-3)
      IF (NET.EQ.1) GO TO 1
      KTYPE(10) = KTYPE(NOSTRA)
C
C DETERMINE THE ORIENTATION AND WRITE IT OUT
C
      IF (NNN.EQ.NOSTRA) KTYPE(NNN) = NALAST
      CALL SYMBOL(6.0,12.5,0.14,'ISOMETRIC VIEW OF',270.0,17)
      CALL SYMBOL(6.0,9.9,0.14,KTYPE(NNN),270.0,4)
      CALL SYMBOL(5.7,12.12,0.14,'VIEWED FROM THE ',270.0,16)
            IF (ZANG.GT.0.0.AND.ZANG.LT.90.0)
     1      CALL SYMBOL(5.7,9.8,0.14,'SE',270.0,2)
            IF (ZANG.GT.90.0.AND.ZANG.LT.180.0)
     2      CALL SYMBOL(5.7,9.8,0.14,'NE',270.0,2)
            IF (ZANG.GT.180.0.AND.ZANG.LT.270.0)
     3      CALL SYMBOL(5.7,9.8,0.14,'NW',270.0,2)
            IF (ZANG.GT.270.0.AND.ZANG.LT.360.0)
     4      CALL SYMBOL(5.7,9.8,0.14,'SW',270.0,2)
      C A L L   N E W P E N (2)
      CALL SYMBOL(4.5,12.9,.14,'CONCEPTUAL DRAWING - NOT TO BE',270.,30)
      CALL SYMBOL(4.1,12.7,0.14,'USED FOR CONTRACT DOCUMENTS',270.0,27)
    1 C A L L   N E W P E N (1)
      C A L L   P L O T   (4.0,0.0,-3)
      IF (XRES.LT.0.01) XRES=0.01
      IF (YRES.LT.1.) YRES=1.
      IF (YRES.GT.100.) YRES=100.
      IF (ELEVA.GT.85.) ELEVA=85.
      IF (ELEVA.LT.-85.)  ELEVA=-85.
      IF (ELEVA.EQ.0.)  ELEVA=0.001
      MATZ=1
      IF (ZANG.LT.0.)  ZANG=360.-ZANG
      IF (ZANG.LT.0.)  R E T U R N
      ANGSAV=ZANG
      A=XMIN
      B=XMAX
      C=YMIN
      D=YMAX
   12 IF (ZANG.LT.90.)  GO TO 15
      IF (ZANG.GT.180.)  GO TO 13
      XMIN=C
      XMAX=D
      YMIN=B
      YMAX=A
      ZANG=ZANG-90.
      MATZ=-1
      GO TO 15
   13 IF (ZANG.GT.270.) GO TO 14
      XMIN=B
      XMAX=A
      YMIN=D
      YMAX=C
```

```
      ZANG=ZANG-180.
      GO TO 15
   14 IF (ZANG.GT.360.) GO TO 16
      XMIN=C
      XMAX=D
      YMIN=A
      YMAX=B
      MATZ=-1
      ZANG=ZANG-270.
      GO TO 15
   16 ZANG=ZANG-360.
      GO TO 12
   15 NA=3
      IF (IPLAN.LT.1)  IAXES=1
      IF (IAXES.LT.1)  IDROP=0
      IF (IPLAN.LT.1)  IDROP=1
      IF (IDROP.GT.0)  NA=2
      ANX=(6.2831853/360.0)*ELEVA
      ANY=(6.2831853/360.0)*ZANG
      IF (ANY.EQ.0.)  ANY=.1E-06
      XLEN=7.*COS(ANY)
      YLEN=7.*SIN(ANY)
      DLZY=YLEN+(SIN(ANX)*(7.-YLEN))
      DLZX=XLEN+(SIN(ANX)*(7.-XLEN))
      NUMX=(7./XRES)+0.1
      NUMY=YRES+1.
      XXX=DLZX**2-XLEN**2
      XPOSX=SQRT(ABS(XXX))
      IF (ELEVA.LT.0.)  XPOSX=-XPOSX
      YYY=DLZY**2-YLEN**2
      XPOSY=SQRT(ABS(YYY))
      IF (ELEVA.LT.0.)  XPOSY=-XPOSY
      DELPOY=YLEN/YRES
      DELPOX=XPOSY/YRES
      DELX=XRES*(XMAX-XMIN)/7.0
      DELY=(YMAX-YMIN)/YRES
C
C SCALE THE PLOT TO FIT ON 18 INCH PAPER
C
      SCALE=(7.0/ABS(XMAX-XMIN))*(COS(ANX))
      XZ=(-1.0)*XPOSX*XRES/7.0
      BX=XLEN*XRES/7.0
      YPOS=(XLEN+YLEN)/2.0+5.5
      C A L L   P L O T   (6.5,-7.,-3)
      C A L L   P L O T   (0.0,3.0,-3)
      C A L L   P L O T   (0.,YPOS,-3)
      XX=XPOSX/(XLEN*100.)
      LL=XLEN*100.+1.1
      ST=(-1.)*XPOSX
      LM=YLEN*100.0+0.5
      YY=XPOSY/(YLEN*100.)
      LM=LM+LL
      IF (IAXES.LT.1) GO TO 4
      XPOS=(-1.)*XPOSX
```

```
      YPOS=XLEN*(-1.)
      C A L L   P L O T   (XPOS,YPOS,2)
      XPOS=XPOS+XPOSY
      YPOS=YPOS-YLEN
      C A L L   P L O T   (XPOS,YPOS,2)
    4 IF (IGRID.GT.0) GO TO 5
C
C TRANSFER BETWEEN LEFT AND RIGHT HANDED VIEWS
C
      IF (ZANG.GT.45.)  GO TO 103
    5 CALL SET(LL,LM,XX,ST,YY,IAXES,NA,XMAX,XMIN,YMAX,YMIN,MATZ,SCALE)
      DO 70 J=1,NUMY
      AY=J-1
      Y=YMIN+DELY*AY
      XPO=DELPOX*AY
      YPO=DELPOY*AY*(-1.)
      X=XMIN
      CALL ZED(X,Y,Z,MATZ)
      XPOS1=XPO+(Z*SCALE)
      YPOS=YPO
C
C BEGIN A NEW SURFICIAL LINE
C
      C A L L   P L O T   (XPOS1,YPOS,3)
      ISTAR=(-100.)*YPO+1.1
      SW=1.
      IF (CK(ISTAR).GT.XPOS1) SW=-1.
      SW2=1.
      IF (CKD(ISTAR).LT.XPOS1)  SW2=-1.
C
C CREATE A VECTOR LINKAGE FOR A SURFICIAL LINE
C
            DO 60 I=1,NUMX
            AY=I
            X=XMIN+(DELX*AY)
            CALL ZED(X,Y,Z,MATZ)
            XPOS2=XPO+(XZ*AY)+(Z*SCALE)
            YNXT=YPO+(BX*AY*(-1.))
            IFRO=YNXT*(-100.)+1.1
            MK=1
            IF (Z.EQ.0.)  MK=IPLAN
            CALL CHECK(SW,SW2,ISTAR,IFRO,XPOS1,XPOS2,YPOS,YNXT,MK)
            YPOS=YNXT
            XPOS1=XPOS2
            ISTAR=IFRO
   60       C O N T I N U E
      IF (IDROP) 70,70,37
   37 XPOS=XPOS1-(Z*SCALE)
      C A L L   P L O T   (XPOS,YPOS,2)
   70 C O N T I N U E
      IF (IGRID.LT.1) GO TO 890
  103 DELX=(XMAX-XMIN)/YRES
      DELY=(YMAX-YMIN)*XRES/7.0
      DELPOY=XLEN/YRES
```

```
      DELPOX=XPOSX/YRES
      XZ=XPOSY*XRES/7.0
      BX=YLEN*XRES/7.0
      CALL SET(LL,LM,XX,ST,YY,IAXES,NA,XMAX,XMIN,YMAX,YMIN,MATZ,SCALE)
            DO 310 J=1,NUMY
            AY=J-1
            X=XMAX-(DELX*AY)
            Y=YMIN
            XPO=DELPOX*AY-XPOSX
            YPO=DELPOY*AY-XLEN
            CALL ZED(X,Y,Z,MATZ)
            IF (IDROP) 104,104,105
105         C A L L   P L O T   (XPO,YPO,3)
104         XPOS1=XPO+(Z*SCALE)
            YPOS=YPO
            C A L L   P L O T   (XPOS1,YPOS,NA)
            ISTAR=(-100.0)*YPO+1.1
            SW=1.0
            IF (CK(ISTAR).GT.XPOS1)  SW=-1.0
            SW2=1.
            IF (CKD(ISTAR).LT.XPOS1)  SW2=-1.
                  DO 210 I=1,NUMX
                  AY=I
                  Y=YMIN+DELY*AY
                  CALL ZED(X,Y,Z,MATZ)
                  XPOS2=XPO+(XZ*AY)+(Z*SCALE)
                  YNXT=YPO+(BX*AY*(-1.0))
                  IFRO=YNXT*(-100.0)+1.1
                  MK=1
                  IF (Z.EQ.0.)  MK=IPLAN
                  CALL CHECK(SW,SW2,ISTAR,IFRO,XPOS1,XPOS2,YPOS,YNXT,MK)
                  YPOS=YNXT
                  XPOS1=XPOS2
                  ISTAR=IFRO
  210             C O N T I N U E
  310       C O N T I N U E
  890 ZLE=0.
      TOP=-100.
C
C DETERMINE THE SIZE OF THE PLOT AND WARN THE USER IF NEEDED
C
            DO 410 I=1,LM
            IF (TOP.LT.CK(I)) TOP=CK(I)
  410       C O N T I N U E
      BOT=0.
      IF (IAXES.GT.0)  GO TO 999
      BOT=100.
            DO 411 I=1,LM
            IF (BOT.GT.CKD(I))  BOT = CKD(I)
  411       C O N T I N U E
  999 ZLE=TOP+0.5-BOT
      IF (MOVE.LT.1)  ZLE=0.
  892 C A L L   P L O T   (ZLE,0.,-3)
      WRITE(6,713) KTYPE(NNN),A,B,C,D,XRES,YRES,ANGSAV,ELEVA,TOP,BOT
```

```
  713 FORMAT('1'/50X,30('*')/2(50X,'*',28X,'*'/),50X,'*',3X,
     A  'THREE DIMENSIONAL PLOT',3X,'*',/,50X,'*',28X,'*',
     B /,50X,'*',12X,A4,12X,'*',/,
     C   2(50X,'*',28X,'*'/),50X,30('*')////10X,'BOUNDRIES (IN INPUT UNI
     DTS)',/15X,'EAST  =',F20.6/15X,'WEST  =',F20.6/
     E  15X,'SOUTH =',F20.6/15X,'NORTH =',F20.6/
     F /15X,'SEPERATION OF THE POINTS =',F5.2,' INCHES'
     G  /15X,'NUMBER OF LINES USED TO FORM PLOT =',F6.0/
     H   15X,'SITE VIEWING ANGLE '
     I,F9.2 /15X,'ANGLE OF ELEVATION OF VIEW =',F9.2/15X,'MAXIMUM DISPLA
     JCEMENT OF PLOTTER PEN IN UPWARD DIRECTION =',F9.2,' INCHES'/
     K15X,'MAXIMUM IN DOWNWARD DIRECTION =',F9.2,' INCHES'///)
      IF ((TOP-BOT).GT.20.)  WRITE(6,714)
      IF (NET.EQ.1) WRITE(LP,715)
  714 FORMAT(10(1X,'IT IS STRONGLY RECOMMENDED THAT YOU DO NOT PLOT THIS
     * DATA SET.  CHECK YOUR DATA - TRY AGAIN'/))
  715 FORMAT(10X,'STACKING FUNCTION WAS ENGAGED',/,10X,29('='),//)
      CALL FACTOR(PZFACT)
      R E T U R N
      E N D
      SUBROUTINE SET(LL,LM,XX,ST,YY,IAXIS,NA,XMAX,XMIN,YMAX,YMIN,MATZ,
     A SCALE)
C ***************************************************************************
C *                                                                         *
C *  SET                                                                    *
C *                                                                         *
C ***************************************************************************
C
C SET CALLS OUT FOR DATA TO CREATE THE PLOT
C
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      IF (IAXIS.LT.1) GO TO 25
      CX=0.
            DO 5 I=1,LL
            CX=CX-1.
            CK(I)=XX*CX
            CKD(I)=-100.
    5       C O N T I N U E
      LL=LL+1
      CY=0.
            DO 10 I=LL,LM
            CY=CY+1.0
            CK(I)=ST+(YY*CY)
            CKD(I)=-100.
   10       C O N T I N U E
      LL=LL-1
      R E T U R N
   25 NA=3
      STEPX=LL
      DELTX=(XMAX-XMIN)/STEPX
      Y=YMIN
      CX=0.
            DO 30 I=1,LL
            X=XMIN+(CX*DELTX)
```

```
            CALL ZED(X,Y,Z,MATZ)
            CX=CX+1.0
            CK(I)=XX*CX*(-1.0)+(Z*SCALE)
            CKD(I)=CK(I)
   30       C O N T I N U E
      LL=LL+1
      STEPY=LM-LL
      DELTY=(YMAX-YMIN)/STEPY
      CY=0.0
      X=XMAX
            DO 35 I=LL,LM
            Y=YMIN+(CY*DELTY)
            CALL ZED(X,Y,Z,MATZ)
            CY=CY+1.0
            CK(I)=ST+(YY*CY)+(Z*SCALE)
            CKD(I)=CK(I)
   35       C O N T I N U E
      LL=LL-1
      R E T U R N
      E N D
      SUBROUTINE ZED(X,Y,Z,K)
C ***************************************************************************
C *                                                                         *
C *  ZED                                                                    *
C *                                                                         *
C ***************************************************************************
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      IF (K) 1,1,2
    1 CALL ZEE(Y,X,Z)
      Z = Z * FACTO
      R E T U R N
    2 CALL ZEE(X,Y,Z)
      Z = Z * FACTO
      R E T U R N
      E N D
      SUBROUTINE ZEE(X,Y,ZFIND)
C ***************************************************************************
C *                                                                         *
C *   ZEE                                                                   *
C *                                                                         *
C ***************************************************************************
C
C ZEE CREATES A SEARCH RADIUS TO FIND THE FIVE NEAREST
C BORINGS.  IT THEN DETERMINES THE ELEVATION OF A POINT IN THE
C PROJECT FROM THESE BORINGS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
```

```
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
      DIMENSION DIST(50),DISTXX(50),SAVE(2),ELEVXX(50)
      RADIUS=SQRT(4.0*((DIFFX)+(DIFFY))/NOPNTS)
      DISMIN=0.1E20
      IF (KEPTBO(NNN).GE.9) N=9
      IF (KEPTBO(NNN).LT.9) N=KEPTBO(NNN)
            DO 4 K=1,NOPNTS
            IF (KPTNME(NNN,K).LT.1) GO TO 4
            DIST(K)= SQRT((X-XCOORD(K))*(X-XCOORD(K))+
     1      (Y-YCOORD(K))*(Y-YCOORD(K)))
            IF (DIST(K).LT.DISMIN) GO TO 60
            GO TO 5
 60         DISMIN=DIST(K)
            LCLOSE=K
 5          C O N T I N U E
            IF (DIST(K).EQ.0.0) GO TO 41
 4          C O N T I N U E
      GO TO 45
 41   ZFIND=ZGIVEN(K)
      R E T U R N
 45   MM=0
            DO 40 KK=1,NOPNTS
            IF (DIST(KK).GT.RADIUS)GO TO 40
            IF (KPTNME(NNN,KK).LT.1) GO TO 40
            MM = MM + 1
            DISTXX(MM)=DIST(KK)
            ELEVXX(MM)=ZGIVEN(KK)
40          C O N T I N U E
      IF (MM.GE.N) GO TO 50
      RADIUS=RADIUS*1.50
      GO TO 45
50    M=MM
100   M=M/2
      IF (M.EQ.0) GO TO 70
      K=MM-M
            DO 400 J=1,K
            I=J
200         L=I+M
            IF (DISTXX(I).LT.DISTXX(L)) GO TO 400
            SAVE(1) = DISTXX(I)
            SAVE(2) = ELEVXX(I)
            DISTXX(I) = DISTXX(L)
            ELEVXX(I) = ELEVXX(L)
            DISTXX(L) = SAVE(1)
            ELEVXX(L) = SAVE(2)
400         C O N T I N U E
      GO TO 100
70    SUM1=0.
      SUM2=0.
C
C THE PROGRAM WILL SEARCH FOR THE NEAREST BORINGS
C
            DO 7 K=1,N
```

```
            SQUARE=DISTXX(K)*DISTXX(K)
            SUM1=SUM1+ELEVXX(K)/SQUARE
            SUM2=SUM2+1./SQUARE
7           C O N T I N U E
      ZFIND=0.5*(ZGIVEN(LCLOSE)+(SUM1/SUM2))
      R E T U R N
      E N D
      BLOCK DATA
C
C INITIALIZE GEPS
C
      COMMON/GEPS1/NAME1(50),XCOORD(52),YCOORD(52),ZGIVEN(52)
      COMMON/GEPS2/XCOORM(52),YCOORM(52),ZGIVEM(52),FACTO
      COMMON/GEPS3/DIFFX,DIFFY,DIFMX,DIFMY,XMIM,YMIM,XMAM,YMAM
      COMMON/GEPS4/XMIN,YMIN,XMAX,YMAX,ZGIVE(50),ZKEEP(50)
      COMMON/GEPS5/ZEEMIN,KEPTBO(10),ELEVA,ZANG,IFLIP1
      COMMON/GEPS6/KR,LP,CK(1100),CKD(1100),NOPNTS,NOBOR
      COMMON/GLBL1/STELEV(10,50),STELEM(10,50),KTYPE(10)
      COMMON/GLBL2/TOTDEP(50),TOTDEM(50),NMXSEC(10),NBXSEC(8)
      COMMON/GLBL3/NMGEPS(10,8),NOXSEC,NOSTRA,CHANGE,NNN,NALAST,NXX
      COMMON/GLBL4/NET,MISIY,MISIX,PZFACT,KPTNME(10,50)
C
C /GEPS1/
C
      DATA NAME1/50*4HGEPS/, XCOORD/52*0./, YCOORD/52*0./, ZGIVEN/52*0./
C
C /GEPS2/
C
      DATA XCOORM/52*0./, YCOORM/52*0./, ZGIVEM/52*0./, FACTO/0./
C
C /GEPS3/
C
      DATA DIFFX/0./, DIFFY/0./, DIFMX/0./, DIFMY/0./, XMIM/0./,
     A YMIM/0./, XMAM/0./, YMAM/0./
C
C /GEPS4/
C
      DATA XMIN/1.E10/, YMIN/1.E10/, XMAX/0./, YMAX/0./, ZGIVE/50*0./,
     A ZKEEP/50*0./
C
C /GEPS5/
C
      DATA ZEEMIN/1.0/, KEPTBO/10*0/, ELEVA/0./, ZANG/0./, IFLIP1/0/
C
C /GEPS6/
C
      DATA KR/5/, LP/6/, CK/1100*0./, CKD/1100*0./, NOPNTS/0/, NOBOR/0/
C
C /GLB1/
C
      DATA STELEV/500*0./, STELEM/500*0./, KTYPE/10*4HSOIL/
C
C /GLBL2/
C
```

```
      DATA TOTDEP/50*0./, TOTDEM/50*0./, NMXSEC/10*0/, NBXSEC/8*0/
C
C /GLBL3/
C
      DATA NMGEPS/80*0/, NOXSEC/0/, NOSTRA/0/, CHANGE/0./, NNN/1/,
     A NALAST/4HLAST/, NXX/0/
C
C /GLBL4/
C
      DATA NET/0/, MISIY/0/, MISIX/0/, PZFACT/1.0/, KPTNME/500*0/
      END
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C **                                                                   **
C **                                                                   **
C **                          S C A N                                  **
C **                          =======                                  **
C **                                                                   **
C **           A FREE FORM INPUT FORTRAN SUBPROGRAM SYSTEM              **
C **                     IBM360  VERSION                               **
C **                                                                   **
C **                                                                   **
C **                                                                   **
C **                                                                   **
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C ***********************************************************************
C *                                                                     *
C * SCAN                                                                *
C *                                                                     *
C ***********************************************************************
      SUBROUTINE SCAN
C
C         SCAN
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
```

```
      COMMON /SCANTB/ NSEPTB, NCLASS, ITAB(256), ICLASS(256)
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      DIMENSION ISTATE(110)
      INTEGER ISTATE, STATE, EXP, OCOL
      EQUIVALENCE(FLT,INT)
      DATA OCOL/0/, NDOT/0/
      DATA ISIG/1/,NSIG/1/,INT/0/,F/.1/,EXP/0/
      DATA IPTLC1/1/, IPTLC2/6/
C                     D  +  -  E  A  Q  .  S  B  $
C                     I           L  U     E  L
C                     G           P  O     P  A
C                     I           H  T        N
C                     T           A  E        K
C        1     GET
C        2     +-
C        3     INTEGER
C        4     REAL
C        5     START EXP
C        6     EXP
C        7     LABEL
C        8     NAME, STRING
C        9     STRING
C        A     ANYTEXT
C        9     ANYTEXT
      DATA ISTATE / 18, 17, 16, 26, 26, 30, 32, 29,  6,  7,
     1              18, 17, 16, 26, 26, 13, 32, 29,  6,  7,
     2              18, 11, 11, 11, 11, 11, 35, 11, 11, 11,
     3              19, 11, 11, 28, 28, 11, 20, 11, 11, 11,
     4              21, 10, 10, 22, 28, 10, 28, 10, 10, 10,
     5              25, 24, 23, 28, 28,  9, 28,  9,  9,  9,
     6              25,  9,  9, 28, 28,  9, 28,  9,  9,  9,
     7               4,  8,  8,  4,  4,  8, 27,  8,  8,  8,
     8               4, 14, 14,  4,  4, 14, 27, 14, 14, 14,
     9               4,  4,  4,  4,  4, 12,  4,  4,  4, 33,
     A               4,  8,  8,  4,  4,  8,  4,  8,  8,  8  /
C
C
C
      NENT=NENT+1
      NCHAR = 0
      STATE = PSTATE
      NBLANK = 0
      ENTITY(1) = BLANK
      JSTART(NENT)=0
      OCOL=COL
      LIM=LIMIT
C
C           GO TO ACTION
C
     1 GO TO (10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,
     1         170,180,190,200,210,220,230,240,250,260,270,280,
     2         290,300,310,320,330,340,350), JUMP
C
C           GET A LINE
```

```
C
    5 DOREAD = .TRUE.
   10 IF(DOREAD)CALL SCANRD
      COL = 0
      NENT = 1
      JSTART(1) = 0
      JUMP=2
      LIM=80
      IF(MODE.EQ.10)GO TO 12
      IF(MODE.EQ.11)GO TO 14
      GO TO 20
C
C             END OF FILE
C
   12 FILPT = FILPT-1
      IF(FILPT.GT.0)INUNIT = FILES(FILPT)
      IF(FILPT.EQ.0)MODE=-10
      IF(LEOF)GO TO 13
      IF(FILPT.GT.0)GO TO 5
      CALL SCANMS(1)
      STOP
   13 JUMP = 1
      DOREAD = .TRUE.
      GO TO 150
C
C              POINT
C
   14 IVAL(1) = 1000*IDIGIT(IPTLC1+0)+100*IDIGIT(IPTLC1+1)+
     1           10*IDIGIT(IPTLC1+2)+    IDIGIT(IPTLC1+3)
      IVAL(2) = 1000*IDIGIT(IPTLC2+0)+100*IDIGIT(IPTLC2+1)+
     1           10*IDIGIT(IPTLC2+2)+    IDIGIT(IPTLC2+3)
      IF(.NOT.MENU)GO TO 16
      IF(SCANMN(IVAL(1),IVAL(2)).LT.0)GO TO 5
      JUMP = 3
      ICOLMN = 1
      COL = 10
      JSTART(1) = 1
      GO TO 150
   16 JSTART(1) = 1
      NCHAR = 10
      COL = 10
      GO TO 80
C
C             NEXT CHARACTER
C
   20 COL = COL+1
C
C             GET CHARACTER
C
   30 JUMP = ISTATE(10*STATE+IBUFF(COL))
      GO TO 1
C
C             SET JUMP TO NEXT CHARACTER
C
```

```
   40 JUMP = 2
C
C             PACK CHARACTER
C
   50 NCHAR = NCHAR + 1
      GO TO 1
C
C             COUNT BLANKS
C
   60 NBLANK = NBLANK + 1
      IF(NBLANK.LE.LIM)GO TO 20
C
C        CHECK IF END OF STRING
C
      IF(PSTATE.EQ.1) GO TO 130
C
C             END LINE
C
   70 IF(.NOT.EOL)GO TO 75
      JUMP = 3
      IF(AUTORD)JUMP = 1
      IF(AUTORD)DOREAD = .TRUE.
      ENTITY(1)=BLANK
      MODE = 9
      NCHAR=0
      NWD = 0
      COL=MAX0(OCOL,1)
      JSTART(NENT)=COL
      GO TO 150
   75 NCHAR = 0
      NWD = 0
      NBLANK = 0
      STATE = PSTATE
      GO TO 5
C
C             PACK BLANK EXIT
C
   80 JUMP = 3
   85 ICOLMN = 0
      IF(NENT.NE.0)ICOLMN = JSTART(NENT)
      CALL SCANPK
      GO TO 150
C
C             END EXPONENT
C
   90 FLT = FLT * 10. ** EXP
      EXP = 0
      NSIG = 1
C
C             END REAL
C
  100 F = 0.1
C
C             END INTEGER
```

```
C
  110 IVALUE = INT
      ISIG = 1
      INT = 0
      GO TO 80
C
C             END STRING
C
  120 SKIP = COL
  340 JUMP = 2
      GO TO 85
C
C             END GET STRING
C
  130 MODE = 8
      PSTATE = 0
      JUMP = 2
      GO TO 150
C
C             END NAME
C
  140 NCHAR = NDOT + 1
      GO TO 80
C
C             RETURN
C
  150 RETURN
C
C             MINUS FOUND
C
  160 ISIG = -1
C
C             PLUS FOUND
C
  170 STATE = 2
      MODE = 6
      INT = ITAB(JBUFF(COL)+1)
      IVALUE = INT
      GO TO 310
C
C             START INTEGER
C
  180 STATE = 3
      INT = 0
      MODE = 1
      IF(JSTART(NENT).EQ.0)JSTART(NENT)=COL
C
C             CONTINUE INTEGER
C
  190 INT = INT*10 + IDIGIT(COL) * ISIG
      GO TO 40
C
C             CHANGE TO REAL
C
```

```
  200 STATE = 4
      MODE = 2
      FLT = INT
      GO TO 40
C
C             CONTINUE REAL
C
  210 FLT = FLT + FLOAT(IDIGIT(COL)*ISIG)*F
      F = F * 0.1
      GO TO 40
C
C             START EXPONENT
C
  220 STATE = 5
      GO TO 40
C
C             MINUS EXP
C
  230 NSIG = -1
C
C             PLUS EXP
C
  240 STATE = 6
      GO TO 40
C
C             CONTINUE EXP
C
  250 EXP = EXP * 10 + IDIGIT(COL)*NSIG
      STATE = 6
      GO TO 40
C
C             START LABEL
C
  260 STATE = 7
      NDOT = 0
      MODE = 3
      GO TO 310
C
C             START NAME
C
  270 STATE = 8
      NDOT = NDOT + 1
      NCHAR = NDOT
      MODE = 4
      JSTART(NENT+NDOT) = COL+1
      GO TO 20
C
C             ANY TEXT
C
  280 MODE = 5
      STATE = 10
      ISIG = 1
      INT = 0
      EXP = 0
```

```
      F = 0.1
      NSIG = 1
      GO TO 40
C
C             SEPARATER
C
  290 INT = ITAB(JBUFF(COL)+1)
      IVALUE = INT
      MODE = 6
      JUMP = 34
      JSTART(NENT) = COL
      ISIG = 1
      GO TO 50
C
C             START STRING
C
  300 STATE = 9
      MODE = 7
  305 JSTART(NENT) = COL
      GO TO 20
C
C             SET BEGINNING COL NUMBER
C
  310 JSTART(NENT) = COL
      IF(MODE.EQ.6.AND.SIGNED)GO TO 290
      GO TO 40
C
C             START REAL
C
  320 JSTART(NENT) = COL
      INT = 0
      GO TO 200
C
C     END LINE AND END STRING
C
  330 SKIP = COL
      GO TO 80
C
C      START SIGNED REAL
C
  350 INT = 0
      GO TO 200
      END
C ************************************************************************
C *                                                                      *
C * RDLINE                                                               *
C *                                                                      *
C ************************************************************************
      SUBROUTINE RDLINE
C
C         READ A CARD
C
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
```

```
     2               IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      JUMP = 1
      CALL SCANRD
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SCANMN                                                                  *
C *                                                                         *
C ***************************************************************************
      INTEGER FUNCTION SCANMN( POINT1, POINT2 )
C
C         MENUING ROUTINE INTERFACE
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      INTEGER POINT1, POINT2
C
C         DUMMY ROUTINE
C
      SCANMN = 0
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SCANMS                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCANMS( ERRNO )
C
C         ISSUE ERROR MESSAGES
C
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      INTEGER ERRNO
C
C         SEND ERROR MESSAGE AND RETURN
C
      GO TO (  10,  20,  30 ), ERRNO
   10 WRITE(IOUT,1001)
      GO TO 9999
   20 WRITE(IOUT,1002)
      GO TO 9999
   30 WRITE(IOUT,1003)
      GO TO 9999
 9999 RETURN
C
 1001 FORMAT(40H0 .....END OF FILE - PROGRAM TERMINATED  /)
 1002 FORMAT(40H0 .....INPUT ERROR - PROGRAM TERMINATED  /)
 1003 FORMAT(40H0 .....FILE LIST OVERFLOW                /)
```

```
      END
C ***************************************************************************
C *                                                                         *
C * SETIN                                                                   *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SETIN(IN)
C
C         SET SCAN INPUT UNIT
C
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      INUNIT = 5
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SETOUT                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SETOUT(OUT)
C
C         SET SCAN ECHO OUTPUT UNIT
C
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      INTEGER OUT
      IOUT = 6
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SETREM                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SETREM(IUNIT)
C
C          SET THE REMOTE UNIT FOR PROMPTING
C
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      LREMOT = IUNIT
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SETFIL                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SETFIL(FILIST,NFILES)
C
C         STACK A LIST OF INPUT FILES
C
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
```

```
      INTEGER FILES, FILPT, FILLIM
      DIMENSION FILIST(10)
      INTEGER FILIST
      NFIL = NFILES
      IF(FILPT+NFIL.LE.FILLIM) GO TO 10
      NFIL=FILLIM-FILPT
      CALL SCANMS(3)
   10 J=FILPT+NFIL+1
      IF(NFIL.EQ.0)GO TO 30
      DO 20 K=1,NFIL
   20 FILES(J-K)=FILIST(K)
      FILPT=FILPT+NFIL
      INUNIT=FILES(FILPT)
   30 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SCINIT                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCINIT(NBLANK, RECLEN, ENDCHR, PROMSW, ECHOSW, COMSW,
     1                  ATRDSW, EOLSW, EOFSW, MENUSW, PTSW, SIGNSW )
C
C         INITIALIZE
C
C
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      LOGICAL PROMSW, ECHOSW, COMSW, ATRDSW, EOLSW, EOFSW, PTSW,
     1        MENUSW, SIGNSW
      INTEGER RECLEN
      INIT = .TRUE.
      ECHAR = ENDCHR
      ECHO = ECHOSW
      LEOF = EOFSW
      EOL = EOLSW
      MENU = MENUSW
      AUTORD = ATRDSW
      COMMNT = .NOT.COMSW
      PROMT = PROMSW
      POINT = PTSW
      SIGNED = SIGNSW
      LIMIT = MIN0(RECSIZ,NBLANK)
      MARK = MIN0(RECSIZ,RECLEN)
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * ADDLAB                                                                  *
C *                                                                         *
```

```
C ***************************************************************************
      SUBROUTINE ADDLAB(LABEL)
C
C         ADD LABEL FOR SCAN ECHO
C
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      ILABEL = LABEL
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * BACKSP                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE BACKSP(NUMENT)
C
C         BACKSPACE THE SCANNER
C
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      IF(NENT.LT.NUMENT) GO TO 10
      NENT = NENT - NUMENT
      COL = JSTART(NENT + 1)
      JUMP = 3
      GO TO 20
   10 COL = 0
      JUMP = 2
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * GETSTR                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE GETSTR
C
C         PARSE A STRING
C
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKLP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COL = JSTART(NENT)
      PSTATE = 1
      JUMP = 2
```

```
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * RESET                                                                   *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE RESET
C
C         START LINE OVER
C
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COL = 0
      JUMP = 2
      NENT = 0
      ENTITY(1) = BLANK
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SKPSTR                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SKPSTR
C
C         SKIP TO END OF STRING
C
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COL = SKIP
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SEPTAB                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SEPTAB(NEWTAB)
C
C         CHANGE SEPERATOR TABLE
C
      DIMENSION NEWTAB(1)
```

```
      COMMON /SCANTB/ NSEPTB, NCLASS, ITAB(256), ICLASS(256)
      DO 10 I = 1, NSEPTB
   10 ITAB(I) = NEWTAB(I)
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * DOSCAN                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION DOSCAN(DUMMY)
C
C          ADVANCE THE SCANNER NOW
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      IF(NEXT)CALL SCAN
      NEXT = .FALSE.
      DOSCAN = .TRUE.
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * ENDCRD                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION ENDCRD(DUMMY)
C
C         IS THE CURRENT SCAN ENTITY AN END OF LINE
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODEOL/9/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 ENDCRD = .FALSE.
      IF(MODE.NE.MODEOL)GO TO 20
      ENDCRD = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * ENDFIL                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION ENDFIL(LAST)
C
C        IS THE CURRENT SCAN ENTITY AN END OF FILE
```

```
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      LOGICAL LAST
      DATA MODEOF/10/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 ENDFIL = .FALSE.
      LAST = .FALSE.
      IF(MODE.NE.IABS(MODEOF))GO TO 20
      ENDFIL = .TRUE.
      IF(MODE.EQ.-MODEOF)LAST = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * ENTIT                                                                   *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE ENTIT(TEXT,NC,NW)
C
C          RETURN THE CURRENT CHARACTERS FROM ENTITY
C
      DIMENSION TEXT(1)
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      NC = NCHAR
      NW = NWD
      DO 10 I = 1, NW
   10 TEXT(I) = ENTITY(I)
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * INTEGR                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION INTEGR(INTEG)
C
C         IS THE CURRENT SCAN ENTITY AN INTEGER
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODINT/1/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
```

```
      NEXT = .FALSE.
   10 INTEGR = .FALSE.
      IF(MODE.NE.MODINT)GO TO 20
      INTEG = IVALUE
      INTEGR = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * LABEL                                                                   *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION LABEL(DUMMY)
C
C         IS THE CURRENT SCAN ENTITY A LABEL
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODLAB/3/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 LABEL = .FALSE.
      IF(MODE.NE.MODLAB)GO TO 20
      LABEL = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * MATCH                                                                   *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION MATCH(STRING,NC)
C
C         MATCH A CHARACTER ARRAY AGAINST THE CURRENT SCAN ENTITY
C
      DIMENSION STRING(1)
      LOGICAL SCANMC
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      MATCH = .FALSE.
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 IF(NCHAR.LT.NC)GO TO 20
      IF(.NOT.SCANMC(STRING(1),ENTITY(1),NC))GO TO 20
      MATCH = .TRUE.
      NEXT = .TRUE.
```

```
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * NAME                                                                    *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION NAME(DUMMY)
C
C         IS THE CURRENT SCAN ENTITY A NAME
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODNAM/4/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 NAME = .FALSE.
      IF(MODE.NE.MODNAM)GO TO 20
      NAME = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * NOSCAN                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION NOSCAN(DUMMY)
C
C         DO NOT ADVANCE THE SCANNER
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      NEXT = .FALSE.
      NOSCAN = .TRUE.
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * NUMI                                                                    *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION NUMI(INTEGR)
C
C         IS THE CURRENT SCAN ENTITY A NUMBER, CONVERT TO INTEGER
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
```

```
      LOGICAL NEXT
      DATA MODREL/2/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 NUMI = .FALSE.
      IF (MODE .GT. MODREL)  GO TO 20
      IF (MODE .EQ. MODREL ) INTEGR = VALUE
      IF (MODE .NE. MODREL ) INTEGR = IVALUE
      NUMI = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * NUMR                                                                    *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION NUMR(REAL)
C
C         IS THE CURRENT SCAN ENTITY A NUMBER, CONVERT TO REAL
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODREL/2/, MODINT/1/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
   10 NEXT = .FALSE.
      NUMR = .FALSE.
      IF (MODE .GT. MODREL) GO TO 20
      IF (MODE .EQ. MODINT ) REAL = IVALUE
      IF (MODE .NE. MODINT ) REAL = VALUE
      NUMR = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * POINT                                                                   *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION POINT(POINT1,POINT2)
C
C         IS THE SCAN ENTITY A POINT
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      INTEGER POINT1, POINT2
      DATA MODPT/11/
      IF(.NOT.NEXT)GO TO 10
```

```
      CALL SCAN
      NEXT = .FALSE.
   10 POINT = .FALSE.
      IF(MODE.NE.MODPT)GO TO 20
      POINT1 = IVAL(1)
      POINT2 = IVAL(2)
      POINT = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * READSC                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE READSC
C
C         ADVANCE THE SCANNER NOW
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      DATA MODECL/9/
      IF(.NOT.AUTORD.OR.(AUTORD.AND.MODE.NE.MODEOL))CALL RDLINE
      NEXT = .TRUE.
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * REALN                                                                   *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION REALN(REAL)
C
C         IS THE CURRENT SCAN ENTITY A REAL
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODREL/2/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 REALN = .FALSE.
      IF(MODE.NE.MODREL)GO TO 20
      REAL = VALUE
```

```
      REALN = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SEP                                                                     *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION SEP(SEPTYP)
C
C         IS THE CURRENT SCAN ENTITY A SEPERATOR
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      INTEGER SEPTYP
      DATA MODSEP/6/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 SEP = .FALSE.
      IF(MODE.NE.MODSEP)GO TO 20
      SEPTYP = IVALUE
      SEP = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * STRING                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION STRING(DUMMY)
C
C         IS THE CURRENT SCAN ENTITY A STRING
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DATA MODSTR/7/
      IF(.NOT.NEXT)GO TO 10
      CALL SCAN
      NEXT = .FALSE.
   10 STRING = .FALSE.
      IF(MODE.NE.MODSTR)GO TO 20
      STRING = .TRUE.
      NEXT = .TRUE.
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
```

```
C * TRUE                                                                *
C *                                                                     *
C ***********************************************************************
      LOGICAL FUNCTION TRUE(DUMMY)
C
C          ADVANCE THE SCANNER BEFORE THE NEXT TEST
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      NEXT = .TRUE.
      TRUE = .TRUE.
      RETURN
      END
C ***********************************************************************
C *                                                                     *
C * INTEGERLIST                                                         *
C *                                                                     *
C ***********************************************************************
      SUBROUTINE TRLIST ( LIST, MLIST, IALL, NLIST, IERR )
C
C           POLO ACTION TO INPUT A LIST OF INTEGER TERMS
C               EACH ACTION MAY BE DELIMITTED BY A COMMA
C               A COMMA PRECEEDING AN EOL INDICATES CONTINUATION
C               TERMS MAY BE:
C                    A) <INTEGER>
C                    B) <INTEGER1> - <INTEGER2>
C                    C) <INTEGER1> TO <INTEGER2>
C                    D) <INTEGER1> - <INTEGER2> BY <INTEGER3>
C                    E) <INTEGER1> TO <INTEGER2> BY <INTEGER3>
C                    F) ALL
C               TYPE A STORES <INTEGER>  IN LIST
C               TYPE B AND C STORES <INTEGER1>, -<INTEGER2>, 1 IN LIST
C               TYPE D AND E STORES <INTEGER1>, -<INTEGER2>, <INTEGER3>
C               IN LIST
C               TYPE F STORES 1, -IALL, 1 IN LIST
C
C          DUMMY ARGUMENTS
C               LIST      (OUTPUT) - LIST OF PARSED INPUT - AS DESCRIBED
C               MLIST     (INPUT)  - ALLOWABLE SIZE OF LIST
C               IALL      (INPUT)  - VALUE OF 'ALL'
C                                    = 0 - 'ALL' IS NOT ACCEPTABLE
C               NLIST     (OUTPUT) - NUMBER OF TERMS STORED IN LIST
C               IERR      (OUTPUT) - ERROR CODE
C                                    = 1 - NO ERROR
C                                    = 2 - PARSE RULES FAILDED
C                                    = 3 - LIST OVERFLOW
C                                    = 4 - LIST NOT FOUND
C
C          COMMON BLOCKS
C               LABELLED COMMON /SCANER/
C
C          CALLED SUBPROGRAMS
```

```
C               SCAN
C               RDLINE
C               SCANMC
C
C          LOCAL VARIABLES
C               ISTATE              - FSA STATES
C               NSTATE              - NEXT STATE TABLE
C               ICLASS              - CLASS OF INPUT
C               IFSA                - ACTION TABLE
C               IACT                - ACTION TO DO
C               IBY                 - HOLLERTH 'BY'
C               ITO                 - HOLLERTH 'TO'
C               JALL                - HOLLERTH 'ALL'
C               ISTART              - FLAG SET TO TRUE ON FIRST SCAN
C               IOVFL               - FLAG SET TO TRUE ON OVERFLOW
C
C          ALGORITHM TERMS
C               STATES    - 1 = START
C                           2 = ITEM (INTEGER)
C                           3 = DELIMITER (,)
C                           4 = ITERATION (-, TO)
C                           5 = INCREMENT (BY)
C                           6 = DELTA (INCREMENT INTEGER)
C               CLASSES   - 1 = INTEGER
C                           2 = ALPHA 'TO' OR SEPERATOR '-'
C                           3 = ALPHA 'BY'
C                           4 = SEPERATOR ','
C                           5 = END OF LINE
C                           6 = ELSE
C               ACTIONS   - Ø = SWITCH STATE
C                           1 = DONE
C                           2 = ERROR
C                           3 = SAVE INGER, - VALUE IMPLIES ITERATION TE
C                           4 = READ LINE
C                           5 = SAVE -1*INTEGER
C                           6 = SAVE 1, SAVE INTEGER
C                           7 = SAVE 1
C                           8 = SAVE 1, DONE
C                           9 = TEST OVERFLOW
C                          1Ø = SAVE ITERATION INCREMENT
C
C
      COMMON /SCANER/ ENTITY(2Ø), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      DIMENSION LIST(MLIST), IFSA(6,6), NSTATE(6,6)
      INTEGER ENTITY
      LOGICAL ISTART, IOVFL
      LOGICAL SCANMC
      DIMENSION IBY(1), ITO(1), JALL(1)
      DATA IBY/2HBY/, ITO/2HTO/, JALL/3HALL/
C
C          TRANSFER TABLE
```

```
C
      DATA IFSA/
     1             3, 1, 1, 0, 1, 1,
     2             3, 9, 2, 0, 1, 1,
     3             3, 2, 2, 2, 4, 2,
     4             5, 2, 2, 2, 2, 2,
     5             6, 2, 0, 7, 8, 8,
     6            10, 2, 2, 0, 4, 2/
C
C          NEXT STATE TABLE
C
      DATA NSTATE/
     1             2, 0, 0, 1, 0, 0,
     2             2, 4, 0, 3, 0, 0,
     3             2, 0, 0, 0, 1, 0,
     4             5, 0, 0, 0, 0, 0,
     5             2, 0, 6, 3, 0, 0,
     6             2, 0, 0, 6, 6, 0/
C
C          INITIALY IN START STATE
C
      ISTART = .TRUE.
      IOVFL = .FALSE.
      NLIST = 1
      ISTATE = 1
C
C          SCAN AND DETERMINE CLASS
C
  100 ICLASS = 6
      IF(.NOT.ISTART)CALL SCAN
      IF(MODE.NE.9)GO TO 110
      ICLASS = 5
      GO TO 140
  110 IF(MODE.NE.6)GO TO 120
      IF(IVALUE.EQ.3)ICLASS = 2
      IF(IVALUE.EQ.16)ICLASS = 4
      GO TO 140
  120 IF(MODE.NE.3)GO TO 130
      IF(SCANMC(ENTITY(1),IBY,2))ICLASS = 3
      IF(SCANMC(ENTITY(1),ITO,2))ICLASS = 2
      IF(SCANMC(ENTITY(1),JALL,3).AND.ISTART.AND.IALL.NE.0)GO TO 350
      GO TO 140
  130 IF(MODE.NE.1)GO TO 140
      ICLASS = 1
C
C          DETERMINE NEXT STATE AND ACTION
C
  140 IACT = IFSA(ICLASS,ISTATE)+1
      ISTATE = NSTATE(ICLASS,ISTATE)
      IF(IACT.NE.2)ISTART=.FALSE.
C
C          ACTION TRANSFER, SKIP STORE ON OVERFLOW
C
      IF(IOVFL.AND.(IACT.EQ.4.OR.IACT.EQ.6.OR.IACT.EQ.7.OR.
```

```
     1              IACT.EQ.8.OR.IACT.EQ.10))GO TO 100
      GO TO (100, 210, 220, 230, 250, 260, 270, 280, 290, 310, 320
     1       ), IACT
C
C         DONE
C
 210  IERR = 1
      NLIST = NLIST-1
      IF(ISTART)IERR = 4
      IF(IOVFL)IERR = 3
      IF(LIST(1).LT.0.AND.NLIST.GT.1)IERR = 2
      RETURN
C
C         ERROR - PARSE
C
 220  IERR = 2
      NLIST = NLIST-1
      RETURN
C
C         SAVE INTEGER, -VALUE BECOMES ITERATION BOUND
C
 230  IF(NLIST.LE.MLIST)GO TO 240
      IOVFL = .TRUE.
      GO TO 100
 240  LIST(NLIST) = IVALUE
      NLIST = NLIST+1
      IF(IVALUE.LT.0.AND.NLIST.EQ.2)GO TO 100
      IF(IVALUE.LT.0)ISTATE = 5
      GO TO 100
C
C         READ LINE
C
 250  CALL RDLINE
      GO TO 100
C
C         SAVE -1*INTEGER
C
 260  LIST(NLIST) = -IVALUE
      IF(IVALUE.LT.0)GO TO 220
      NLIST = NLIST+1
      GO TO 100
C
C         SAVE 1, SAVE INTEGER
C
 270  LIST(NLIST) = 1
      LIST(NLIST+1) = IVALUE
      NLIST = NLIST+2
      GO TO 100
C
C         SAVE 1
C
 280  LIST(NLIST) = 1
      NLIST = NLIST+1
      GO TO 100
```

```
C
C         SAVE 1, DONE
C
 290  IF(IOVFL)GO TO 300
      LIST(NLIST) = 1
 300  IERR = 1
      IF(IOVFL)IERR = 3
      IF(LIST(1).LT.0.AND.NLIST.NE.1)IERR = 2
      RETURN
C
C         TEST FOR OVERFLOW IN ITERATION TERM
C
 310  IF(NLIST+1.GT.MLIST)IOVFL = .TRUE.
      GO TO 100
C
C         SAVE INTEGER FOR INCREMENT
C
 320  IF(NLIST.LE.MLIST)GO TO 330
      IOVFL = .TRUE.
      GO TO 100
 330  LIST(NLIST) = IVALUE
      NLIST = NLIST+1
      GO TO 100
C
C        ALL, DONE
C
 350  IF(MLIST.LT.3)GO TO 360
      LIST(1) = 1
      LIST(2) = -IALL
      LIST(3) = 1
      NLIST = 3
      IERR = 1
      CALL SCAN
      IF(MODE.NE.6)RETURN
      IF(IVALUE.NE.16)RETURN
      CALL SCAN
      IF(MODE.NE.9)RETURN
      CALL RDLINE
      CALL SCAN
      RETURN
 360  IERR = 3
      NLIST = 0
      RETURN
C
      END
C ***************************************************************************
C *                                                                         *
C * ALPHA MATCH                                                             *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE TRAMAT ( WORD, MCHAR, MATCHL, LOC, SCNACT )
C
C         ALPHAMATCH
C         MATCH A WORD AGAINST A LIST OF WORDS
```

```
C
C         DUMMY ARGUMENTS
C               WORD      (INPUT)  - WORD TO MATCH AGAINST LIST
C               MCHAR     (INPUT)  - LENGTH OF WORD IN CHARACTERS
C               MATCHL    (INPUT)  - LIST OF WORDS TO MATCH AGAINST
C                                    FIRST ENTRY IS THE NUMBER OF WORDS
C                                    THE WORD ENTRIES FOLLOW, EACH BEGINS
C                                    WITH ONE WORD WHICH IS THE LENGTH
C               LOC       (OUTPUT) - THE LOCATION OF WORD IN MATCHL
C                                    = 0 IF NOT FOUND
C               SCNACT    (INPUT)  - SCAN ACTION IF MATCH IS FOUND
C                                    = 1 - SCAN
C                                    = 2 - SCAN & SKIP ,
C                                    = 3 - SCAN & SKIP , & END
C                                          DO THE READLINE BUT DON'T SCAN
C                                    = 4 - SCAN & SKIP , & END
C                                          READLINE & SCAN
C                                    = ELSE - NOSCAN
C
C
      DIMENSION WORD(1), MATCHL(1)
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     *1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      INTEGER SCNACT
      LOGICAL SCANMC
C
C         GO THROUGH THE LIST TO FIND THE MATCH
C
      N = MATCHL(1)
      LOC = 0
      J = 2
      DO 10 I = 1, N
      L = MATCHL(J)
      IF(L.GT.NCHAR)GO TO 10
      IF(SCANMC(WORD,MATCHL(J+1),L))GO TO 20
   10 J = J+(L+NCPW-1)/NCPW+1
      RETURN
   20 LOC = I
      IF(SCNACT.LT.1.OR.SCNACT.GT.4)RETURN
      CALL SCAN
      IF(SCNACT.EQ.1)RETURN
      IF(MODE.NE.6.OR.IVALUE.NE.16)RETURN
      CALL SCAN
      IF(SCNACT.EQ.2)RETURN
      IF(MODE.NE.9)RETURN
      CALL RDLINE
      IF(SCNACT.EQ.3)RETURN
      CALL SCAN
      RETURN
      END
C ***************************************************************************
```

```
C *                                                                    *
C * ALPHA LIST                                                         *
C *                                                                    *
C **********************************************************************
      SUBROUTINE TRALST ( MATCHL, IFOUND, NMAX, NFOUND, IERROR )
C
C         ALPHA LIST
C         MATCH ALPHA INPUT AGAINST A LIST OF WORDS
C
C         DUMMY ARGUMENTS
C               MATCHL   (INPUT)  - LIST OF WORDS TO MATCH AGAINST
C                                   FIRST ENTRY IS THE NUMBER OF WORDS
C                                   THE WORD ENTRIES FOLLOW, EACH BEGINS
C                                   WITH ONE WORD WHICH IS THE LENGTH
C               IFOUND   (OUTPUT) - A VECTOR OF WHAT MATCHES WERE FOUND
C               NMAX     (INPUT)  - THE LENGTH OF IFOUND
C               NFOUND   (OUTPUT) - THE NUMBER OF MATCHES, ENTRIES IN
C                                   IFOUND
C               IERROR   (OUTPUT) - ERROR FLAG
C                                   = 1 - NO ERRORS
C                                   = 2 = SYNTAX
C                                   = 3 - OVERFLOW OF IFOUND
C
C
      DIMENSION MATCHL(1), IFOUND(1)
      LOGICAL SCANMC
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
C
C         GO THROUGH THE LIST TO LOOK FOR A MATCH
C
      IERROR = 1
      NFOUND = 0
      N = MATCHL(1)
      IF(MODE.EQ.3)GO TO 10
      RETURN
   10 J = 2
      DO 20 I = 1, N
      L = MATCHL(J)
      IF(L.GT.NCHAR)GO TO 20
      IF(SCANMC(ENTITY(1),MATCHL(J+1),L))GO TO 30
   20 J = J+(L+NCPW-1)/NCPW+1
      IF(NFOUND.GT.NMAX)IERROR = 3
      RETURN
C
C         SAVE IT AND GET THE NEXT ITEM
C         SKIP , OR AUTOREAD ON ,&EOL
C
   30 NFOUND = NFOUND+1
      IF(NFOUND.LE.NMAX)IFOUND(NFOUND) = I
      CALL SCAN
```

```
      IF(MODE.EQ.3)GO TO 10
      IF(MODE.EQ.6.AND.IVALUE.EQ.16)GO TO 50
      IF(NFOUND.GT.NMAX)IERROR = 3
      RETURN
   50 CALL SCAN
      IF(MODE.EQ.9)GO TO 60
      IF(MODE.EQ.3)GO TO 10
      IERROR = 2
      RETURN
   60 CALL RDLINE
      CALL SCAN
      IF(MODE.EQ.3)GO TO 10
      IF(NFOUND.GT.NMAX)IERROR = 3
      RETURN
      END
C ***********************************************************************
C *                                                                     *
C * NEXT ON INTEGER LIST                                                *
C *                                                                     *
C ***********************************************************************
      SUBROUTINE TRXLST ( LIST, NL, IPLIST, ICN, NEXT )
C
C         POLO ACTION TO DETERMINE NEXT ENTRY ON AN INTEGERLIST
C
C         DUMMY ARGUMENTS
C              LIST      (INPUT)  - LIST OF DATA
C              NL        (INPUT)  - LENGTH OF LIST
C              IPLIST    (UPDATE) - INPUT  - CURRENT LIST ELEMENT
C                                           - = 1 ON FIRST CALL
C                                   OUTPUT - NEXT LIST ELEMENT
C                                            - = Ø ON END OF LIST
C                                              NEXT IS RETURNED
C              ICN       (UPDATE) - INPUT  - CURRENT ITERATION COUNT
C                                            - = Ø ON FIRST CALL
C                                   OUTPUT - NEXT ITERATION COUNT
C              NEXT      (OUTPUT) - NEXT ITEM ON LIST TO BE PROCCESSED
C
C
      DIMENSION LIST(NL)
C
C         GET RESULT
C
      IF(IPLIST.LE.Ø)RETURN
      NEXT = LIST(IPLIST)
      IPLIST = IPLIST+1
      IF(IPLIST.GT.NL)GO TO 120
C
C         NEXT IS PLUS, SINGLE TERM
C
      IF(LIST(IPLIST).GE.Ø)RETURN
C
C         ITERATION TERM (ALWAYS DONE ONCE), LOOP EXHAUSTED
C
      NEXT = NEXT+ICN*LIST(IPLIST+1)
```

```
      IR = NEXT+LIST(IPLIST+1)
      IF(LIST(IPLIST+1).GT.0.AND.
     1  (IR.LE.IABS(LIST(IPLIST))))
     2  GO TO 110
      IF(LIST(IPLIST+1).LT.0.AND.
     1  (IR.GE.IABS(LIST(IPLIST))))
     2  GO TO 110
      ICN =  0
      IPLIST = IPLIST+2
      IF(IPLIST.GT.NL)GO TO 120
      RETURN
C
C         ITERATE
C
 110  IPLIST = IPLIST-1
      ICN = ICN+1
      RETURN
C
C         END OF LIST
C
 120  IPLIST = 0
      RETURN
      END
C ***********************************************************************
C *                                                                     *
C * ISZLST - SIZE OF A LIST                                             *
C *                                                                     *
C ***********************************************************************
      INTEGER FUNCTION ISZLST ( LIST, NL )
C
C
C          DETERMINE THE NUMBER OF TERMS IN AN INTEGERLIST
C
C
      DIMENSION LIST(NL)
      IP = 0
      IR = 0
   10 IF ( IP.GE.NL ) GO TO 20
      IP = IP + 1
      IR = IR + 1
      IF ( LIST(IP).GE.0 ) GO TO 10
      IR = IR + ((IABS(LIST(IP))-LIST(IP-1)+LIST(IP+1))/LIST(IP+1))-2
      IP = IP + 1
      GO TO 10
   20 ISZLST = IR
      RETURN
      END
C ***********************************************************************
C *                                                                     *
C * SCANDB                                                              *
C *                                                                     *
C ***********************************************************************
      SUBROUTINE SCANDB
C
```

```
C          DEBUG - ALL CHARACTER OUTPUT AT 4 PER WORD
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      WRITE(IOUT,1001)
      WRITE(IOUT,1002)RECSIZ, LIMIT, MARK, ECHAR, ECHO, PROMT, ILABEL,
     1                INIT, POINT, MENU, LEOF, EOL, AUTORD, COMMNT,
     2                SIGNED
      WRITE(IOUT,1003)INUNIT, IOUT, LREMOT, FILPT, FILLIM, FILES
      WRITE(IOUT,1004)COL, JUMP, NENT, PSTATE, SKIP, DOREAD, INCOL,
     3                CARD, JBUFF, IBUFF, IDIGIT
      WRITE(IOUT,1005)ENTITY, MODE, VALUE, IVALUE, IVAL, NCHAR, NWD,
     1                NEXT, ICOLMN
      RETURN
 1001 FORMAT(1H1/24H ************************/
     1           24H * S C A N   D E B U G */
     2           24H ************************/)
 1002 FORMAT( 5X, 25HC O N T R O L   S T A T E/
     1       10X, 14HRECORD SIZE - , I2/
     2       10X, 15HRECORD LIMIT - , I2/
     3       10X, 13HRECORD END - , I2/
     4       10X, 23HRECORD END CHARACTER - , A1/
     5       10X, 14HECHO SWITCH - , L1/
     6       10X, 16HPROMPT SWITCH - , L1/
     7       10X, 13HLINE LABEL - , I5/
     8       10X, 24HINITIALIZATION SWITCH - , L1/
     9       10X, 15HPOINT SWITCH - , L1/
     A       10X, 14HMENU SWITCH - , L1/
     B       10X, 21HEND OF FILE SWITCH - , L1/
     C       10X, 21HEND OF LINE SWITCH - , L1/
     D       10X, 19HAUTO READ SWITCH - ,L1/
     E       10X, 17HCOMMENT SWITCH - , L1/
     F       10X, 14HSIGN SWITCH - , L1/)
 1003 FORMAT( 5X, 17HI / O   S T A T E/
     1       10X, 13HINPUT UNIT - , I2/
     2       10X, 14HOUTPUT UNIT - , I2/
     3       10X, 14HREMOTE UNIT - , I2/
     4       10X, 21HFILE STACK POINTER - , I2/
     5       10X, 13HFILE LIMIT - , I2/
```

```
     6       10X, 13HFILE STACK - , 10(I2, 2X)/)
 1004 FORMAT( 5X, 19HS C A N   S T A T E/
     1       10X, 9HCOLUMN - , I2/
     2       10X, 7HJUMP - , I2/
     3       10X, 16HENTITY NUMBER - , I2/
     4       10X, 15HSTRING STATE - , I2/
     5       10X, 13HSKIP STATE - , I2/
     6       10X, 14HREAD SWITCH - , L1/
     7       10X, 14HRECORD SIZE - , I2/
     8       10X, 13HINPUT LINE - , 81A1/
     9       10X, 18HINTERNAL RECORD - , 20(I3,1X)/28X,20(I3,1X)/
     A            28X,20(I3,1X)/28X,21(I3,1X)/
     B       10X, 8HCLASS - , 20(I3,1X)/18X,20(I3,1X)/18X,20(I3,1X)/
     C            18X,21(I3,1X)/
     D       10X, 9HDIGITS - , 20(I3,1X)/19X,20(I3,1X)/19X,20(I3,1X)/
     E            19X,21(I3,1X)/)
C
C                THE A4 FORMAT IS MACHINE DEPENDENT
C
 1005 FORMAT( 5X, 11HE N T I T Y/
     1       10X, 9HENTITY - , 20A4/
     2       10X, 6HVALUE - , G20.10/
     3       10X, 9HIVALUE - , I20/
     4       10X, 7HIVAL - , 2I20/
     5       10X, 7HMODE - , I2/
     6       10X, 12HCHARACTER - , I2/
     7       10X, 8HWORDS - , I2/
     8       10X, 7HNEXT - , L1/
     9       10X, 9HCOLUMN - , I2///)
      END
C ***************************************************************************
C *                                                                         *
C * SCANCL                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCANCL
C
C       CLASSIFY THE INPUT CHARACTERS
C
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COMMON /SCANTB/ NSEPTB, NCLASS, ITAB(256), ICLASS(256)
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      LOGICAL*1 XCARD(320), XBUFF(320)
      EQUIVALENCE ( XCARD(1), CARD(1) ), ( XBUFF(1), JBUFF(1) )
```

```
      J = 1
      K = NCPW
      NN = MIN0(INCOL,MARK)
      NN = NN+1
      CARD(NN) = ECHAR
      DO 10 I = 1, NN
      XBUFF(K) = XCARD(J)
      IDIGIT(I) = -1
      IF(JBUFF(I).GE.INTZER.AND.JBUFF(I).LT.INTNIN)
     1    IDIGIT(I) = JBUFF(I)-INTZER
      J = J+NCPW
      K = K+NCPW
      IBUFF(I) = ICLASS(JBUFF(I)+1)
      IF(CARD(I).EQ.ECHAR)IBUFF(I) = 10
   10 CONTINUE
      RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SCANIN                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCANIN(IN,BUFF,RECLEN,ERRCOD)
C
C         INPUT A RECORD FROM A DEVICE
C
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      INTEGER RECLEN, ERRCOD, RECSIZ
      DIMENSION BUFF(80)
      RECSIZ = RECLEN
      ERRCOD = 0
      READ(IN,1001,ERR=20,END=30)(BUFF(I),I=1,RECSIZ)
      DO 10 I = 1, RECSIZ
      IF(BUFF(RECLEN).NE.BLANK)GO TO 40
   10 RECLEN = RECLEN-1
      GO TO 40
   20 ERRCOD = 1
      GO TO 40
   30 ERRCOD = -1
   40 RECLEN = MAX0(RECLEN,2)
      RETURN
 1001 FORMAT(80A1)
      END
C ***************************************************************************
C *                                                                         *
C * SCANPK                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCANPK
```

```
C
C           PACK THE SCANER RESULT IN ENTITY AT 4 CHARACTERS PER WORD
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      LOGICAL*1 XENTIT(80), XCARD(320)
      EQUIVALENCE ( XENTIT(1), ENTITY(1) ), ( XCARD(1), CARD(1) )
      NWD = (NCHAR+NCPW-1)/NCPW
      IF(NWD.LE.0)NWD = 1
      ENTITY(NWD) = BLANK
      IF(NWD.LT.20)ENTITY(NWD+1) = BLANK
      ISTART = (ICOLMN-1)*NCPW+1
      IF(NCHAR.EQ.0)NWD = 0
      IF(NCHAR.EQ.0)GO TO 20
      IF(MODE.EQ.7)ISTART = ISTART+NCPW
      DO 10 I = 1, NCHAR
      XENTIT(I) = XCARD(ISTART)
      ISTART = ISTART+NCPW
      IF(MODE.EQ.4)ISTART = (JSTART(NENT+I)-1)*NCPW+1
   10 CONTINUE
   20 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * SCANRD                                                                  *
C *                                                                         *
C ***************************************************************************
      SUBROUTINE SCANRD
C
C         INPUT FOR SCAN
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
     1                POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2                AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1                INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2                IDIGIT(81), CARD(81)
```

```
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      LOGICAL REREAD
C
C         READ A RECORD
C
      DOREAD = .FALSE.
   10 IF(PROMT)WRITE(LREMOT,1003)
      REREAD = .FALSE.
      INCOL = RECSIZ
      CALL SCANIN(INUNIT,CARD,INCOL,IERR)
      IF(IERR)20,40,30
C
C         EOF,ERR
C
   20 MODE = 10
      GO TO 100
   30 CALL SCANMS(2)
      STOP
C
C         CLASSIFY
C
   40 CALL SCANCL
      MODE = 0
C
C         COMMENT  'C ' IN CC 1&2  = ZC3, Z40 =  195, 64
C
      IF(JBUFF(1).NE.INTC.OR.JBUFF(2).NE.INTBLK)GO TO 50
      IF(COMMNT)REREAD = .TRUE.
      GO TO 70
C
C         POINT 'DDDD,DDDD' IN CC1-9 = (4@ZF0-F9),Z6B,(4@ZF0-F9)
C                                    = (4@240-249),107,(4@240-249)
C
   50 IF(.NOT.POINT)GO TO 70
      IF(INCOL.NE.9)GO TO 70
      IF(JBUFF(5).NE.INTCOM)GO TO 70
      DO 60 I = 1, 4
      IF(IDIGIT(I).EQ.-1.OR.IDIGIT(I+5).EQ.-1)GO TO 70
   60 CONTINUE
      MODE = 11
      GO TO 70
C
C         ECHO (NOT A POINT WITH MENUING)
C
   70 IF(MODE.EQ.11.AND.MENU)GO TO 100
      IF(.NOT.ECHO)GO TO 90
      NN = MIN0(INCOL,MARK)
      IF(ILABEL.EQ.0)GO TO 80
      WRITE(IOUT,1001)ILABEL,(CARD(I),I=1,NN)
      ILABEL = 0
      GO TO 90
   80 WRITE(IOUT,1002)(CARD(I),I=1,NN)
```

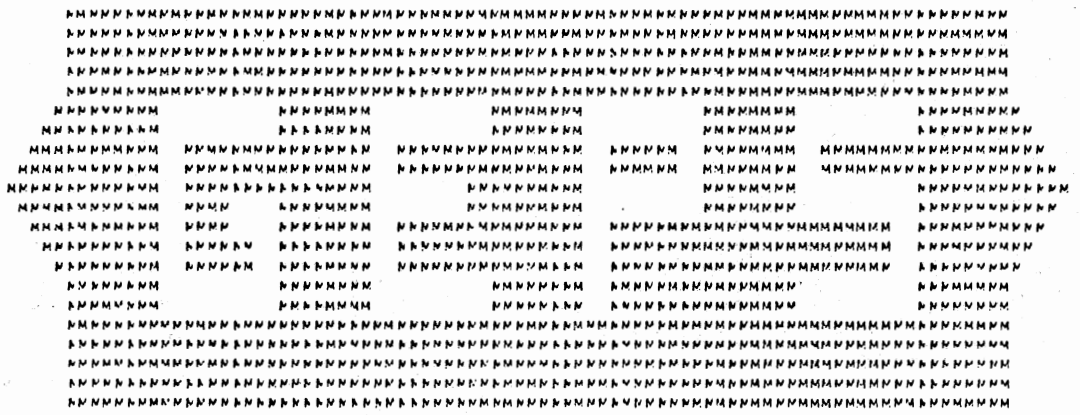
```
   90 IF(REREAD)GO TO 10
  100 RETURN
 1001 FORMAT(1X,I5,1X,80A1)
 1002 FORMAT(7X,      80A1)
 1003 FORMAT(3H ? )
      END
C ***************************************************************************
C *                                                                         *
C * SCANMC                                                                  *
C *                                                                         *
C ***************************************************************************
      LOGICAL FUNCTION SCANMC(TEXTA,TEXTB,NCHAR)
C
C         CHARACTER MATCH OF A AND B FOR NCHARS
C
      DIMENSION TEXTA(1), TEXTB(1)
      REAL JA, JB, JC, JD
      LOGICAL*1 AA(4), BB(4), CC(4), DD(4)
      EQUIVALENCE (AA(1),JA), (BB(1),JB), (CC(1),JC), (DD(1),JD)
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
      SCANMC = .FALSE.
      NW = NCHAR/NCPW
      NR = NCHAR-NW*NCPW
      IF(NW.EQ.0)GO TO 20
      DO 10 I = 1, NW
      IF(TEXTA(I).NE.TEXTB(I))GO TO 50
   10 CONTINUE
   20 IF(NR.EQ.0)GO TO 40
      JA = TEXTA(NW+1)
      JB = TEXTB(NW+1)
      JC = BLANK
      JD = BLANK
      DO 30 I = 1, NR
      CC(4) = AA(I)
      DD(4) = BB(I)
      IF(JC.NE.JD)GO TO 50
   30 CONTINUE
   40 SCANMC = .TRUE.
   50 RETURN
      END
C ***************************************************************************
C *                                                                         *
C * BLOCK DATA                                                              *
C *                                                                         *
C ***************************************************************************
       BLOCK DATA
C
C           INITIALZE THE SCANNER
C
      COMMON /SCANER/ ENTITY(20), IVAL(2), MODE, NCHAR, NWD, NEXT,
     1                ICOLMN
      EQUIVALENCE (IVAL(1),VALUE,IVALUE)
      LOGICAL NEXT
      COMMON /SCANCT/ ECHAR, ECHO, ILABEL, LIMIT, MARK, PROMT,
```

```
     1               POINT, RECSIZ, INIT, LEOF, EOL, MENU,
     2               AUTORD, COMMNT, SIGNED
      INTEGER RECSIZ
      LOGICAL ECHO, PROMT, POINT, INIT, LEOF, EOL, MENU,
     1        COMMNT, AUTORD, SIGNED
      COMMON /SCANIO/ INUNIT, IOUT, FILES(10), FILPT, FILLIM, LREMOT
      INTEGER FILES, FILPT, FILLIM
      COMMON /SCANLN/ COL, JUMP, NENT, PSTATE, SKIP, DOREAD,
     1               INCOL, JSTART(80), IBUFF(81), JBUFF(81),
     2               IDIGIT(81), CARD(81)
      INTEGER COL, SKIP, PSTATE
      LOGICAL DOREAD
      COMMON /SCANTB/ NSEPTB, NCLASS, ITAB(256), ICLASS(256)
      COMMON /SCANIM/ NCPW, BLANK, INTZER, INTNIN, INTC, INTCOM, INTBLK
C
C         /SCANER/
C
      DATA ENTITY/20*Z40404040/, IVAL/2*0/, MODE/0/, NCHAR/0/,
     1     NWD/0/, NEXT/.FALSE./, ICOLMN/0/
C
C         /SCANCT/
C
      DATA ECHAR/1H$/, ECHO/.TRUE./, ILABEL/0/, LIMIT/80/, MARK/80/,
     1     POINT/.FALSE./, RECSIZ/80/, INIT/.FALSE./, LEOF/.FALSE./,
     2     EOL/.TRUE./, MENU/.FALSE./, AUTORD/.FALSE./,
     3     COMMNT/.TRUE./, SIGNED/.FALSE./, PROMT/.FALSE./
C
C         /SCANIO/
C
      DATA INUNIT/5/, IOUT/6/, FILES/10*0/, FILPT/0/, FILLIM/10/,
     1     LREMOT/6/
C
C         /SCANLN/
C
      DATA COL/0/, JUMP/1/, NENT/1/, PSTATE/0/, SKIP/1/,
     1     DOREAD/.TRUE./, INCOL/80/, JSTART/80*0/,
     2     IBUFF/81*9/, JBUFF/81*9/, IDIGIT/81*0/,
     3     CARD/81*Z40404040/
C
C         /SCANTB/
C
      DATA NSEPTB/256/, NCLASS/256/
      DATA ITAB/
     1      74*0,1,2,14,4,5,6,7,9*0,
     2      8,9,10,11,12,13,3,15,9*0,16,17,18,19,20,
     3      10*0,21,22,23,24,25,26,128*0/
      DATA ICLASS/64*8,
     1           9,10*8,7,2*8,2,8,
     2           11*8,8,4*8,
     3           3,12*8,5,8,8,
     4           13*8,6,8,6,
     1           9*5,7*8,   9*5,7*8,   8,8*5,7*8,   16*8,
     5           8,4*5,4,4*5,6*8,
     6           8,9*5,6*8,
```

```
     7             8,8,8*5,6*8,
     8             10*1,6*8/
C
C         /SCANIM/
C
      DATA NCPW/4/, BLANK/Z40404040/, INTZER/240/,
     1     INTNIN/250/, INTC/195/, INTCOM/107/, INTBLK/64/
C
      END
```

APPENDIX B

GLOMOD EXAMPLE LISTING



```
GEOTECHNICAL
ENGINEERING
PLOTTING
SYSTEM

OKLAHOMA
STATE
UNIVERSITY

LEVEL II

MAY 1981
```

```
C ***************************************************************      00022390
C *                                                             *      00022400
C *    DATA INPUT BLOCK I                                       *      00022410
C *                                                             *      00022420
C ***************************************************************      00022430
C                                                                      00022440
C                                                                      00022450
C JOB 4813 (8-72)  PROJECT -VLB-  PROVIDENCE STATION, RHODE ISLAND     00022460
C RELOCATION AND PARKING GARAGE   1/4/80 TO 2/11/80                    00022470
C                                                                      00022480
C                                                                      00022490
 ENGAGE GLGMOD                                                         00022500
C                                                                      00022510
C                                                                      00022520
  COMMANDS     NOGRID  CROP  SCLIC  MOVE                               00022530
  STATISTICS   GENERATION 0.1  LINES  100                              00022540
  NUMBER OF BORINGS   20                                               00022550
  IMPERIAL                                                             00022560
C ***************************************************************      00022570
C *                                                             *      00022580
C *   DATA INPUT BLOCK II                                       *      00022590
C *                                                             *      00022600
C ***************************************************************      00022610
 HOLE   01    ELEVATION 12.5 LATITUDE 1771.7 LONGITUDE 390.5 DEPTH 26.5 00022620
  H    02     E 14.4 LAT 1880.9 LON 482.4 D 33.0                       00022630
  H    03     E 17.7 LAT 1932.0 LON 841.1 D 36.5                       00022640
  H    04     E 21.9 LAT 2071.2 LON 1067.2 D 26.5                      00022650
  H    05     E 14.7 LAT 2061.1 LON 877.2 D 69.0                       00022660
  H    06     E 14.1 LAT 2170.5 LON 957.3 D 41.5                       00022670
  H    07     E 13.7 LAT 2031.1 LON 646.2 D 66.5                       00022680
  H    08     E 12.5 LAT 2145.8 LON 774.3 D 63.0                       00022690
  H    09     E 13.5 LAT 2248.3 LON 885.5 D 50.0                       00022700
  H    010    E 11.0 LAT 2133.8 LON 474.2 D 46.5                       00022710
  H    011    E 11.0 LAT 2267.1 LON 620.8 D 71.2                       00022720
  H    012    E 13.7 LAT 2387.1 LON 758.5 D 46.5                       00022730
  H    013    E 11.4 LAT 2363.4 LON 481.6 D 31.5                       00022740
  H    014    E 11.0 LAT 2259.6 LON 745.4 D 64.3                       00022750
  H    015    E 11.8 LAT 2141.2 LON 624.8 D 66.6                       00022760
  H    016    E 15.1 LAT 2222.2 LON 1184.6 D 21.5                      00022770
  H    017    E 15.9 LAT 1866.1 LON 661.3 D 31.5                       00022780
  H    018    E 11.7 LAT 1528.3 LON 321.6 D 26.5                       00022790
  H    019    E 28.1 LAT 1897.4 LON 913.8 D 46.5                       00022800
  H    020    E 17.4 LAT 1807.0 LON 725.7 D 30.2                       00022810
C ***************************************************************      00022820
C *                                                             *      00022830
C *   DATA INPUT BLOCK III                                      *      00022840
C *                                                             *      00022850
C ***************************************************************      00022860
  NUMBER OF STRATA 6                                                   00022870
  FILL                                                                 00022880
    01      20.5                                                       00022890
    02      19.0                                                       00022900
    03      23.0                                                       00022910
    04      22.5                                                       00022920
    05      18.0                                                       00022930
    06      14.5                                                       00022940
    07      18.0                                                       00022950
    08      15.0                                                       00022960
```

```
   U9      16.5                                                00022970
   D10     17.0                                                00022980
   D11     17.3                                                00022990
   D12     19.0                                                00023000
   D13     14.0                                                00023010
   U14     14.5                                                00023020
   D15     14.5                                                00023030
   D16     17.0                                                00023040
   D17     19.0                                                00023050
   D18     16.5                                                00023060
   D19     29.5                                                00023070
   U20     24.0                                                00023080
URGI                                                           00023090
D1  25.0                                                       00023100
P2  24.5                                                       00023110
D3  32.0                                                       00023120
D4  24.5                                                       00023130
UP5  25.0                                                      00023140
D6  18.0                                                       00023150
U7  22.5                                                       00023160
U8  25.5                                                       00023170
U9  21.0                                                       00023180
D10  23.0                                                      00023190
U11  22.0                                                      00023200
D12  27.5                                                      00023210
D13  25.0                                                      00023220
U14  24.0                                                      00023230
D15  24.0                                                      00023240
D16  NOT                                                       00023250
D17  27.5                                                      00023260
D18  22.7                                                      00023270
D19  NOT                                                       00023280
U20  29.0                                                      00023290
SAND                                                           00023300
D1  NOT                                                        00023310
P2  NOT                                                        00023320
D3  NOT                                                        00023330
D4  NOT                                                        00023340
UP5  33.0                                                      00023350
U6  28.5                                                       00023360
U7  32.5                                                       00023370
U8  36.5                                                       00023380
U9  23.5                                                       00023390
D10  29.0                                                      00023400
U11  37.7                                                      00023410
D12  35.5                                                      00023420
D13  NOT                                                       00023430
U14  37.0                                                      00023440
D15  34.5                                                      00023450
D16  NOT                                                       00023460
D17  NOT                                                       00023470
D18  NOT                                                       00023480
D19  39.0                                                      00023490
U20  NOT                                                       00023500
SILT                                                           00023510
D1  NOT                                                        00023520
P2  NOT                                                        00023530
```

```
D3  NOT                                                    00023540
D4  NOT                                                    00023550
DP5  48.0                                                  00023560
D6  34.4                                                   00023570
D7  41.0                                                   00023580
D8  53.0                                                   00023590
D9  36.3                                                   00023600
D10  43.0                                                  00023610
D11  51.0                                                  00023620
D12  44.0                                                  00023630
D13  NOT                                                   00023640
D14  51.0                                                  00023650
D15  49.0                                                  00023660
D16  NOT                                                   00023670
D17  NOT                                                   00023680
D19  NOT                                                   00023700
D20  NOT                                                   00023710
TILL                                                       00023720
D1  NOT                                                    00023730
P2  NOT                                                    00023740
D3  NOT                                                    00023750
D4  NOT                                                    00023760
DP5  69.0                                                  00023770
D6  NOT                                                    00023780
D7  66.5                                                   00023790
D8  73.0                                                   00023800
D9  NOT                                                    00023810
D10  NOT                                                   00023820
D11  71.5                                                  00023830
D12  NOT                                                   00023840
D13  NOT                                                   00023850
D14  54.2                                                  00023860
D15  66.5                                                  00023870
D16  NOT                                                   00023880
D17  NOT                                                   00023890
D18  NOT                                                   00023900
D19  NOT                                                   00023910
D20  NOT                                                   00023920
DATA                                                       00023930
D1  9.5                                                    00023940
P2  9.8                                                    00023950
D3  NOT                                                    00023960
D4  NOT                                                    00023970
DP5  7.9                                                   00023980
D6  NOT                                                    00023990
D7  NOT                                                    00024000
D8  9.0                                                    00024010
D9  NOT                                                    00024020
D10  12.6                                                  00024030
D11  NOT                                                   00024040
D12  12.0                                                  00024050
D13  NOT                                                   00024060
D14  NOT                                                   00024070
D15  NOT                                                   00024080
```

```
  D16  NOT                                                              00024090
  D17  NOT                                                              00024100
  D18  NOT                                                              00024110
  D19  27.0                                                             00024120
  D20  11.0                                                             00024130
C ************************************************************************00024140
C *                                                                    *  00024150
C *   DATA INPUT BLOCK IV                                              *  00024160
C *                                                                    *  00024170
C ************************************************************************00024180
  NUMBER OF SECTIONS  5                                                 00024190
XSEC ONEP   NUMBER OF ECRINGS IN SECTION  4  ECRINGS ARE  15 17 9 16     00024200
XSEC TWOP   NUMBER OF ECRINGS IN SECTION  4  ECRINGS ARE  5 8 11 13      00024210
XSEC THRP   NUMBER OF ECRINGS IN SECTION  4  ECRINGS ARE  4 6 9 12       00024220
XSEC FOUP   NUMBER OF ECRINGS IN SECTION  3  ECRINGS ARE  10 11 12       00024230
XSEC FIVP   NUMBER OF ECRINGS IN SECTION  4  ECRINGS ARE  17 7 15 11     00024240
  VIEWING ELEVATION  35.0                                               00024250
  VIEWING ANGLE      45.0                                               00024260
  FINAL STRATA   ROCK                                                   00024270


. . . . END OF INPUT EXECUTION + + +
```

<<<<<<< RUN ANALYSIS >>>>>>>

THIS PROBLEM FILLED 44% OF GEPS MAXIMUM SPACE.

THE APPROXIMATE LENGTH OF THE PLOT IS 200 INCHES.

THIS PROBLEM HAD A TOTAL BORING FOOTAGE OF 915.3 FEET.

ESTIMATED TIME TO CONSTRUCT THE PLOT IS 85 TO 95 MINUTES.

BORING LOCATION PLAN

BORING	LATITUDE FT/KM		LONGITUDE FT/KM		ELEVATION FEET/MTRS	
D1	1771.70/	0.5400	350.80/	0.1190	12.50/	3.8100
P2	1880.90/	0.5733	488.40/	0.1489	14.40/	4.3891
C3	1932.00/	0.5889	841.10/	0.2564	17.70/	5.3950
C4	2071.20/	0.6313	1067.30/	0.3253	21.90/	6.6751
UP5	2061.10/	0.6282	877.20/	0.2674	14.70/	4.4806
D6	2170.50/	0.6616	957.30/	0.2918	14.10/	4.2977
U7	2031.10/	0.6191	648.20/	0.1976	13.70/	4.1758
U8	2145.80/	0.6540	774.30/	0.2360	12.50/	3.8100
U9	2248.30/	0.6853	885.50/	0.2699	13.50/	4.1148
D10	2133.80/	0.6504	474.20/	0.1445	11.00/	3.3528
U11	2267.10/	0.6910	620.80/	0.1892	11.00/	3.3528
D12	2387.10/	0.7276	765.50/	0.2333	13.70/	4.1758
D13	2363.40/	0.7204	481.60/	0.1468	11.40/	3.4747
U14	2259.60/	0.6887	749.40/	0.2284	11.00/	3.3528
D15	2141.20/	0.6526	624.80/	0.1904	11.80/	3.5967
D16	2222.20/	0.6773	1184.60/	0.3611	15.10/	4.6025
D17	1866.10/	0.5688	661.30/	0.2016	15.90/	4.8463
D18	1528.30/	0.4659	321.80/	0.0981	11.70/	3.5662
D19	1897.40/	0.5783	913.80/	0.2785	28.10/	8.5649
U20	1807.00/	0.5508	725.70/	0.2212	17.40/	5.3035

CROSS SECTIONS

BORING	GEPS NAME	TOTAL DEPTH FEET/MTRS	NUMBER OF SOILS ENCOUNTERED
D1	1	26.500/ 8.077	3
P2	2	33.000/ 10.058	3
D3	3	36.500/ 11.125	2
D4	4	26.500/ 8.077	2
UP5	5	69.000/ 21.031	6
D6	6	41.500/ 12.649	4
U7	7	66.500/ 20.269	5
U8	8	83.000/ 25.298	6
U9	9	50.000/ 15.240	4
D10	10	46.500/ 14.173	5
U11	11	71.200/ 21.702	5
D12	12	46.500/ 14.173	5
D13	13	31.500/ 9.601	2
U14	14	64.300/ 19.599	5
D15	15	66.600/ 20.300	5
D16	16	21.500/ 6.553	1
D17	17	31.500/ 9.601	2
D18	18	26.500/ 8.077	2
D19	19	46.500/ 14.173	3
U20	20	30.200/ 9.205	3

STRATA DESIGNATION

---- FILL ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
D1	20.500/ 6.248	20.500/ 6.248	-8.000/ -2.438
P2	19.000/ 5.791	19.000/ 5.791	-4.600/ -1.402
D3	23.000/ 7.010	23.000/ 7.010	-8.300/ -1.615
D4	22.500/ 6.858	22.500/ 6.858	-0.600/ -0.183
UP5	18.000/ 5.486	18.000/ 5.486	-3.300/ -1.006
D6	14.500/ 4.420	14.500/ 4.420	-0.400/ -0.122
U7	18.000/ 5.486	18.000/ 5.486	-4.300/ -1.311
U8	15.000/ 4.572	15.000/ 4.572	-2.500/ -0.762
U9	16.500/ 5.029	16.500/ 5.029	-3.000/ -0.914
D10	17.000/ 5.182	17.000/ 5.182	-6.000/ -1.829
U11	17.300/ 5.273	17.300/ 5.273	-6.300/ -1.920
D12	19.000/ 5.791	19.000/ 5.791	-5.300/ -1.615
D13	14.000/ 4.267	14.000/ 4.267	-2.600/ -0.792
U14	14.500/ 4.420	14.500/ 4.420	-3.500/ -1.067
D15	14.500/ 4.420	14.500/ 4.420	-2.700/ -0.823
D16	17.000/ 5.182	17.000/ 5.182	-1.500/ -0.579
D17	19.000/ 5.791	19.000/ 5.791	-3.100/ -0.945
D18	18.500/ 5.639	18.500/ 5.639	-6.800/ -2.073
D19	29.500/ 8.992	29.500/ 8.992	-1.400/ -0.427
U20	24.000/ 7.315	24.000/ 7.315	-6.600/ -2.012

STRATA DESIGNATION

---- ORGI ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
D1	25.000/ 7.620	4.500/ 1.372	-12.500/ -3.810
P2	24.500/ 7.468	5.500/ 1.676	-10.100/ -3.078
D3	32.000/ 9.754	9.000/ 2.743	-14.300/ -4.359
D4	24.500/ 7.468	2.000/ 0.610	-2.600/ -0.792
UP5	25.000/ 7.620	7.000/ 2.134	-10.300/ -3.139
D6	18.000/ 5.486	3.500/ 1.067	-3.900/ -1.189
U7	22.500/ 6.858	4.500/ 1.372	-8.800/ -2.682
U8	25.500/ 7.772	10.500/ 3.200	-13.000/ -3.962
U9	21.000/ 6.401	4.500/ 1.372	-7.500/ -2.286
D10	23.000/ 7.010	6.000/ 1.829	-12.000/ -3.658
U11	22.000/ 6.706	4.700/ 1.433	-11.000/ -3.353
D12	27.500/ 8.382	8.500/ 2.591	-13.600/ -4.206
D13	25.000/ 7.620	11.000/ 3.353	-13.600/ -4.145
U14	24.000/ 7.315	9.500/ 2.896	-13.000/ -3.962
D15	24.000/ 7.315	9.500/ 2.896	-12.200/ -3.719
D17	27.500/ 8.382	8.500/ 2.591	-11.600/ -3.536
D18	22.700/ 6.919	4.200/ 1.280	-11.000/ -3.353
U20	25.000/ 8.839	5.000/ 1.524	-11.600/ -3.536

STRATA DESIGNATION

---- SAND ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
UP5	33.000/ 10.058	8.000/ 2.438	-18.300/ -5.578
D6	28.500/ 8.687	10.500/ 3.200	-14.400/ -4.389
U7	32.500/ 9.906	10.000/ 3.048	-18.800/ -5.730
U8	36.500/ 11.125	11.000/ 3.353	-24.000/ -7.315
U9	28.500/ 8.687	7.500/ 2.286	-15.000/ -4.572
D10	29.000/ 8.839	6.000/ 1.829	-18.000/ -5.486
U11	37.700/ 11.491	15.700/ 4.785	-26.700/ -8.138
D12	35.500/ 10.820	8.000/ 2.438	-21.800/ -6.645
U14	37.000/ 11.278	13.000/ 3.962	-26.000/ -7.925
D15	34.500/ 10.516	10.500/ 3.200	-22.700/ -6.919
D19	39.000/ 11.887	39.000/ 11.887	-10.900/ -3.322

STRATA DESIGNATION

---- SILT ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
UP5	48.000/ 14.630	15.000/ 4.572	-33.300/-10.150
D6	34.400/ 10.485	5.900/ 1.798	-20.300/ -6.187
U7	41.000/ 12.497	8.500/ 2.591	-27.300/ -8.321
U8	53.000/ 16.154	16.500/ 5.029	-40.500/-12.344
U9	36.300/ 11.064	7.800/ 2.377	-22.800/ -6.949
D10	43.000/ 13.106	14.000/ 4.267	-32.000/ -9.754
U11	51.000/ 15.545	13.300/ 4.054	-40.000/-12.192
D12	44.000/ 13.411	8.500/ 2.591	-30.300/ -9.235
U14	51.000/ 15.545	14.000/ 4.267	-40.000/-12.192
D15	49.000/ 14.935	14.500/ 4.420	-37.200/-11.339

STRATA DESIGNATION

---- TILL ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
UP5	69.000/ 21.031	21.000/ 6.401	-54.300/-16.551
U7	66.500/ 20.269	25.500/ 7.772	-52.000/-16.093
U8	73.000/ 22.250	20.000/ 6.096	-60.500/-18.440
U11	71.000/ 21.641	20.000/ 6.096	-60.000/-18.288
U14	64.200/ 19.568	13.200/ 4.023	-53.200/-16.215
D15	66.500/ 20.269	17.500/ 5.334	-54.700/-16.673

STRATA DESIGNATION

---- WATA ----

BORING	DEPTH TO BOTTOM OF STRATA FEET/MTRS	THICKNESS OF STRATA FEET/MTRS	ELEVATION OF STRATA FEET/MTRS
D1	9.500/ 2.896	9.500/ 2.896	3.000/ 0.914
P2	9.800/ 2.987	9.800/ 2.987	4.600/ 1.402
UP5	7.900/ 2.408	7.900/ 2.408	6.800/ 2.073
U8	9.000/ 2.743	9.000/ 2.743	3.500/ 1.067
D10	12.600/ 3.840	12.600/ 3.840	-1.600/ -0.488
D12	12.000/ 3.658	12.000/ 3.658	1.700/ 0.518
D19	27.000/ 8.230	27.000/ 8.230	1.100/ 0.335
U20	11.500/ 3.505	11.500/ 3.505	5.900/ 1.798

CROSS SECTION

---- NUMBER ONEP ----

GEPS NAME	LATITUDE DISTANCE FT/KM	LONGITUDE DISTANCE FT/KM	STRAIGHT LINE DISTANCE FT/KM
18			
	337.80/ 0.103	339.50/ 0.103	478.92/ 0.146
17			
	195.00/ 0.059	215.90/ 0.066	290.93/ 0.089
5			
	161.10/ 0.049	307.40/ 0.094	347.06/ 0.106
16			

CROSS SECTION

---- NUMBER TWOP ----

GEPS NAME	LATITUDE DISTANCE FT/KM	LONGITUDE DISTANCE FT/KM	STRAIGHT LINE DISTANCE FT/KM
5			
	84.70/ 0.026	-102.90/-0.031	133.28/ 0.041
8			
	121.30/ 0.037	-153.50/-0.047	195.64/ 0.060
11			
	96.30/ 0.029	-139.20/-0.042	169.26/ 0.052
13			

CROSS SECTION

---- NUMBER THRU ----

GEPS NAME	LATITUDE DISTANCE FT/KM	LONGITUDE DISTANCE FT/KM	STRAIGHT LINE DISTANCE FT/KM
4			
	99.30/ 0.030	-110.00/-0.034	148.19/ 0.045
6			
	77.80/ 0.024	-71.80/-0.022	105.87/ 0.032
9			
	138.80/ 0.042	-126.60/-0.039	187.86/ 0.057
12			

CROSS SECTION

---- NUMBER FOUR ----

GEPS NAME	LATITUDE DISTANCE FT/KM	LONGITUDE DISTANCE FT/KM	STRAIGHT LINE DISTANCE FT/KM
10			
	133.30/ 0.041	146.60/ 0.045	198.14/ 0.060
11			
	120.00/ 0.037	138.10/ 0.042	182.95/ 0.056
12			

CROSS SECTION

---- NUMBER FIVE ----

GEPS NAME	LATITUDE DISTANCE FT/KM	LONGITUDE DISTANCE FT/KM	STRAIGHT LINE DISTANCE FT/KM
17			
	165.00/ 0.050	-13.10/-0.004	165.52/ 0.050
7			
	110.10/ 0.034	-23.40/-0.007	112.56/ 0.034
15			
	125.90/ 0.038	-4.00/-0.001	125.96/ 0.038
11			

```
**********************************
*                                *
*                                *
*   THREE DIMENSIONAL PLOT       *
*                                *
*          FILL                  *
*                                *
*                                *
**********************************
```

```
BOUNDRIES (IN INPUT UNITS)
     EAST  =          1529.29980
     WEST  =          2387.09985
     SOUTH =           321.79980E05
     NORTH =          1184.59985

     SEPERATION OF THE POINTS = 0.10 INCHES
     NUMBER OF LINES USED TO FORM PLOT =  100.
     SITE VIEWING ANGLE      45.00
     ANGLE OF ELEVATION OF VIEW =     35.00
     MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION =     11.47 INCHES
     MAXIMUM IN DOWNWARD DIRECTION =      0.0  INCHES
```

```
**********************************
*                                *
*                                *
*   THREE DIMENSIONAL PLOT       *
*                                *
*          CRGI                  *
*                                *
*                                *
**********************************
```

```
BOUNDRIES (IN INPUT UNITS)
     EAST  =          1528.29560
     WEST  =          2367.09585
     SOUTH =           321.79560E08
     NORTH =          1184.89585

     SEPERATION OF THE POINTS = 0.10 INCHES
     NUMBER OF LINES USED TO FORM PLOT =  100.
     SITE VIEWING ANGLE     45.00
     ANGLE OF ELEVATION OF VIEW =    35.00
     MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION =      8.46 INCHES
     MAXIMUM IN DOWNWARD DIRECTION =      0.0  INCHES
```

```
*****************************
*                           *
*                           *
*   THREE DIMENSIONAL PLOT  *
*                           *
*          SAND             *
*                           *
*                           *
*****************************
```

```
BOUNDRIES (IN INPUT UNITS)
     EAST  =          1528.29560
     WEST  =          2387.09565
     SOUTH =           321.75560E05
     NORTH =          1057.29960

     SEPERATION OF THE POINTS = 0.10 INCHES
     NUMBER OF LINES USED TO FORM PLOT =  100
     SITE VIEWING ANGLE     45.00
     ANGLE OF ELEVATION OF VIEW =    35.00
     MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION =      8.43 INCHES
     MAXIMUM IN DOWNWARD DIRECTION =     0.0  INCHES
```

```
**********************************
*                                *
*                                *
*   THREE DIMENSIONAL PLOT       *
*                                *
*            SILT                *
*                                *
*                                *
**********************************
```

```
BOUNDRIES (IN INPUT UNITS)
     EAST  =          1857.39990
     WEST  =          2387.09985
     SOUTH =           474.19951
     NORTH =           957.29905

     SEPERATION OF THE POINTS = 0.10 INCHES
     NUMBER OF LINES USED TO FORM PLOT =  100.
     SITE VIEWING ANGLE      45.00
     ANGLE OF ELEVATION OF VIEW =    35.00
     MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION =      5.65 INCHES
     MAXIMUM IN DOWNWARD DIRECTION =      0.0  INCHES
```

BOUNDRIES (IN INPUT UNITS)
EAST = 2031.095E5
WEST = 2387.095E5
SOUTH = 474.195E51
NORTH = 957.295E05

SEPERATION OF THE POINTS = 0.10 INCHES
NUMBER OF LINES USED TO FORM PLOT = 100
SITE VIEWING ANGLE 45.00
ANGLE OF ELEVATION OF VIEW = 35.00
MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION = 4.74 INCHES
MAXIMUM IN DOWNWARD DIRECTION = 0.0 INCHES

BOUNDRIES (IN INPUT UNITS)
EAST = 2031.C9918
WEST = 2267.C9965
SOUTH = 620.799E05
NORTH = 877.199951

SEPERATION OF THE POINTS = 0.10 INCHES
NUMBER OF LINES USED TO FORM PLOT = 100.
SIDE VIEWING ANGLE 45.00
ANGLE OF ELEVATION OF VIEW = 35.00
MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION = 4.44 INCHES
MAXIMUM IN DOWNWARD DIRECTION = 0.0 INCHES

CONTOUR GRID FOR FILL STRATA

	ND 1	ND 2	ND 3	ND 4	ND 5	ND 6	ND 7	ND 8	ND 9	ND10	ND11	ND12	ND13	ND14	ND15	ND16	ND17	ND18	ND19	ND20	ND21	ND22	ND23	ND24	ND25
PT25	23	23	23	23	23	23	23	23	23	20	20	20	20	20	20	20	20	16	15	15	15	15	15	15	15
PT24	23	23	23	23	23	23	23	23	23	23	20	20	20	20	20	21	20	20	16	15	15	15	15	15	15
PT23	23	23	23	23	23	23	23	24	24	24	24	20	20	20	21	21	21	20	19	15	15	15	15	15	14
PT22	23	23	23	23	23	23	24	24	24	24	24	24	20	20	21	21	21	20	19	15	14	15	15	14	14
PT21	23	23	23	23	23	23	24	24	24	24	24	24	24	20	20	20	20	19	15	14	14	14	14	14	13
PT20	23	23	23	23	23	24	24	24	24	25	25	25	24	23	20	20	19	14	14	14	14	14	13	13	13
PT19	17	18	23	23	23	23	24	24	25	26	26	26	25	23	16	15	15	14	14	14	14	13	13	13	13
PT18	17	17	17	18	23	23	24	24	25	26	27	27	25	23	15	15	15	14	14	14	13	13	13	13	13
PT17	17	17	17	18	18	18	23	24	24	25	26	25	18	18	15	14	14	14	14	13	13	13	13	13	13
PT16	17	17	17	17	17	18	18	18	24	24	18	18	17	17	15	14	14	14	13	13	13	13	13	13	13
PT15	17	17	17	17	17	17	17	17	18	18	18	17	17	17	15	14	13	12	12	12	11	11	11	13	13
PT14	17	17	17	17	17	17	17	17	17	17	17	17	17	17	16	13	12	12	12	12	11	11	11	13	13
PT13	17	17	17	17	17	17	17	17	17	17	17	16	17	14	14	13	12	12	12	12	11	11	11	13	13
PT12	16	16	16	17	17	17	17	17	17	17	16	16	15	14	14	13	13	12	12	11	11	11	11	12	13
PT11	16	16	16	16	16	17	17	17	17	16	15	15	15	14	13	13	13	11	11	11	11	11	11	11	12
PT10	16	16	16	16	16	16	16	16	16	15	15	15	14	13	13	13	12	11	11	11	11	11	11	11	11
PT 9	13	16	16	16	16	16	15	15	15	15	15	15	14	13	13	13	12	11	11	11	11	11	11	11	11
PT 8	13	14	13	13	13	13	15	14	14	14	14	14	14	13	13	13	12	11	11	11	11	11	11	11	11
PT 7	13	13	13	13	13	13	13	14	14	14	14	14	14	13	12	11	11	11	11	11	11	11	11	11	11
PT 6	12	12	12	13	13	13	13	13	14	14	14	14	14	13	11	11	11	11	11	11	11	11	11	11	11
PT 5	12	12	12	13	13	12	12	12	14	14	14	14	14	13	11	11	11	11	11	11	11	11	11	11	11
PT 4	12	12	12	12	12	12	12	12	12	14	14	14	14	13	11	11	11	11	11	11	11	11	11	11	11
PT 3	11	12	12	12	12	12	12	12	12	12	14	13	12	13	11	11	11	11	11	11	11	11	11	11	11
PT 2	11	11	11	12	12	12	12	12	12	12	12	13	13	13	11	11	11	11	11	11	11	11	11	11	11
PT 1	11	11	11	12	12	12	12	12	12	12	12	15	13	13	11	11	11	11	11	11	11	11	11	11	11

COORDINATE LEGEND
FOR
FILL STRATUM

	LATITUDE			LONGITUDE	
NODE	1	1528	POINT	1	321
NODE	2	1563	POINT	2	356
NODE	3	1598	POINT	3	391
NODE	4	1633	POINT	4	426
NODE	5	1668	POINT	5	461
NODE	6	1703	POINT	6	496
NODE	7	1738	POINT	7	531
NODE	8	1773	POINT	8	566
NODE	9	1808	POINT	9	601
NODE	10	1843	POINT	10	636
NODE	11	1878	POINT	11	671
NODE	12	1913	POINT	12	706
NODE	13	1948	POINT	13	741
NODE	14	1983	POINT	14	776
NODE	15	2018	POINT	15	811
NODE	16	2053	POINT	16	846
NODE	17	2088	POINT	17	881
NODE	18	2123	POINT	18	916
NODE	19	2158	POINT	19	951
NODE	20	2193	POINT	20	986
NODE	21	2228	POINT	21	1021
NODE	22	2263	POINT	22	1056
NODE	23	2298	POINT	23	1091
NODE	24	2333	POINT	24	1126
NODE	25	2368	POINT	25	1161

CONTOUR GRID FOR ORGI STRATA

	ND 1	ND 2	ND 3	ND 4	ND 5	ND 6	ND 7	ND 8	ND 9	ND10	ND11	ND12	ND13	ND14	ND15	ND16	ND17	ND18	ND19	ND20	ND21	ND22	ND23	ND24	ND25
PT25	-10	-10	-10	-12	-12	-12	-11	-11	-11	-5	-5	-5	-4	-4	-3	-2	-2	-3	-3	-4	-5	-5	-5	-6	-8
PT24	-10	-10	-10	-10	-12	-12	-12	-12	-11	-11	-5	-5	-5	-4	-3	-3	-3	-3	-4	-4	-5	-5	-5	-7	-8
PT23	-10	-10	-11	-11	-12	-12	-12	-12	-12	-12	-11	-11	-5	-5	-4	-4	-4	-4	-4	-4	-5	-5	-7	-7	-8
PT22	-10	-10	-11	-10	-10	-12	-12	-12	-12	-12	-12	-12	-11	-9	-9	-8	-5	-4	-4	-4	-5	-7	-7	-8	-8
PT21	-11	-10	-11	-11	-11	-11	-12	-12	-12	-12	-12	-12	-12	-10	-9	-9	-9	-5	-4	-4	-7	-7	-7	-8	-8
PT20	-11	-11	-11	-11	-11	-11	-11	-12	-12	-12	-13	-13	-12	-10	-10	-10	-9	-5	-5	-5	-6	-7	-7	-7	-8	-8
PT19	-11	-11	-11	-11	-11	-11	-11	-11	-12	-13	-13	-13	-13	-12	-10	-10	-10	-9	-6	-7	-7	-7	-9	-8	-8
PT18	-11	-11	-11	-11	-11	-11	-11	-11	-11	-13	-13	-14	-14	-13	-10	-10	-10	-10	-11	-8	-8	-8	-8	-8	-12
PT17	-11	-11	-11	-11	-11	-11	-11	-11	-11	-13	-13	-13	-13	-13	-10	-10	-10	-12	-12	-11	-8	-11	-11	-12	-13
PT16	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-13	-13	-13	-13	-10	-10	-12	-12	-12	-12	-12	-12	-12	-13	-13
PT15	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-13	-13	-12	-9	-12	-12	-12	-12	-12	-12	-12	-12	-13	-13
PT14	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-9	-9	-9	-12	-12	-12	-12	-12	-12	-12	-13	-13
PT13	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-9	-9	-9	-9	-11	-12	-12	-12	-12	-12	-13	-13
PT12	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-9	-8	-9	-9	-11	-12	-12	-11	-11	-11	-11	-13
PT11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-9	-9	-8	-9	-11	-12	-12	-12	-11	-11	-11	-11	-11
PT10	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-11	-9	-9	-9	-9	-11	-12	-12	-11	-11	-11	-11	-11	-11
PT 9	-11	-11	-12	-12	-11	-11	-11	-11	-10	-10	-10	-10	-10	-9	-8	-9	-11	-11	-12	-11	-11	-11	-11	-11	-13
PT 8	-11	-11	-11	-12	-12	-11	-10	-10	-10	-10	-10	-10	-10	-10	-9	-11	-11	-11	-11	-11	-11	-11	-11	-12	-13	-13
PT 7	-11	-11	-11	-12	-12	-12	-11	-10	-10	-10	-10	-10	-10	-10	-11	-11	-11	-11	-11	-11	-11	-11	-12	-13	-13	-13
PT 6	-11	-11	-11	-12	-12	-12	-12	-12	-10	-10	-10	-10	-10	-10	-11	-11	-11	-11	-11	-11	-11	-12	-12	-13	-13	-13
PT 5	-11	-11	-11	-12	-12	-12	-12	-12	-12	-10	-10	-10	-10	-10	-11	-11	-11	-11	-11	-11	-11	-12	-13	-13	-13	-13
PT 4	-11	-11	-11	-11	-12	-12	-12	-12	-12	-11	-10	-10	-10	-10	-11	-11	-11	-11	-11	-11	-12	-12	-12	-13	-13	-13
PT 3	-11	-11	-11	-11	-12	-12	-12	-12	-12	-12	-10	-10	-10	-10	-11	-11	-11	-11	-11	-11	-11	-12	-12	-13	-13	-13
PT 2	-11	-11	-11	-11	-12	-12	-12	-12	-12	-12	-11	-10	-10	-10	-11	-11	-11	-11	-11	-11	-12	-12	-12	-13	-13	-13
PT 1	-11	-11	-11	-11	-12	-12	-12	-12	-12	-12	-12	-11	-10	-10	-11	-11	-11	-11	-11	-11	-12	-12	-12	-13	-13	-13

COORDINATE LEGEND
FOR
CRGI STRATUM

	LATITUDE			LONGITUDE	
NODE	1	1528	POINT	1	321
NODE	2	1563	POINT	2	352
NODE	3	1598	POINT	3	383
NODE	4	1633	POINT	4	414
NODE	5	1668	POINT	5	445
NODE	6	1703	POINT	6	476
NODE	7	1738	POINT	7	507
NODE	8	1773	POINT	8	538
NODE	9	1808	POINT	9	569
NODE	10	1843	POINT	10	600
NODE	11	1878	POINT	11	631
NODE	12	1913	POINT	12	662
NODE	13	1948	POINT	13	693
NODE	14	1983	POINT	14	724
NODE	15	2018	POINT	15	755
NODE	16	2053	POINT	16	786
NODE	17	2088	POINT	17	817
NODE	18	2123	POINT	18	848
NODE	19	2158	POINT	19	879
NODE	20	2193	POINT	20	910
NODE	21	2228	POINT	21	941
NODE	22	2263	POINT	22	972
NODE	23	2298	POINT	23	1003
NODE	24	2333	POINT	24	1034
NODE	25	2368	POINT	25	1065

CONTOUR GRID
FOR
SAND STRATA

	NO 1	NO 2	NO 3	NO 4	NO 5	NO 6	NO 7	NO 8	NO 9	NO10	NO11	NO12	NO13	NO14	NO15	NO16	NO17	NO18	NO19	NO20	NO21	NO22	NO23	NO24	NO25
PT25	-11	-11	-12	-12	-13	-17	-17	-17	-17	-17	-16	-15	-14	-14	-14	-14	-15	-15	-15	-15	-16	-16	-16	-16	-17
PT24	-11	-11	-11	-12	-13	-17	-17	-17	-17	-17	-17	-15	-15	-14	-14	-14	-15	-15	-15	-15	-16	-16	-16	-16	-17
PT23	-10	-11	-11	-12	-13	-17	-17	-16	-16	-16	-16	-17	-15	-15	-15	-15	-15	-15	-15	-15	-16	-16	-16	-17	-17
PT22	-11	-11	-11	-12	-13	-17	-16	-16	-16	-16	-16	-16	-15	-15	-15	-15	-15	-15	-15	-15	-16	-16	-16	-17	-17
PT21	-11	-11	-12	-12	-17	-17	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-15	-15	-15	-15	-16	-16	-17	-17	-21
PT20	-12	-12	-12	-13	-17	-17	-16	-16	-16	-16	-16	-16	-16	-21	-17	-16	-16	-15	-15	-16	-16	-17	-17	-21	-21
PT19	-12	-12	-13	-13	-17	-16	-16	-16	-16	-16	-16	-21	-21	-22	-22	-17	-17	-16	-16	-17	-17	-17	-21	-21	-21
PT18	-13	-13	-13	-14	-16	-16	-16	-16	-16	-19	-22	-23	-23	-23	-22	-22	-23	-23	-23	-23	-23	-23	-21	-21	-21
PT17	-13	-13	-14	-16	-16	-16	-16	-16	-19	-22	-22	-23	-23	-23	-23	-23	-24	-24	-24	-24	-24	-24	-21	-21	-21
PT16	-13	-14	-14	-16	-16	-16	-16	-19	-22	-22	-23	-23	-23	-23	-23	-23	-24	-25	-25	-25	-24	-24	-22	-21	-21
PT15	-14	-16	-16	-16	-16	-19	-19	-19	-22	-22	-23	-23	-23	-23	-23	-23	-25	-25	-25	-25	-25	-24	-22	-21	-21
PT14	-16	-16	-16	-16	-16	-19	-19	-19	-19	-22	-23	-23	-23	-23	-23	-24	-25	-25	-25	-25	-26	-24	-22	-22	-21
PT13	-16	-16	-16	-16	-19	-19	-19	-19	-19	-19	-22	-23	-23	-23	-23	-24	-25	-25	-25	-25	-25	-24	-22	-22	-22
PT12	-16	-19	-19	-19	-19	-19	-19	-19	-19	-19	-20	-23	-23	-22	-22	-24	-24	-25	-25	-25	-25	-24	-22	-22	-22
PT11	-19	-19	-19	-19	-19	-19	-18	-18	-19	-19	-22	-23	-23	-22	-22	-23	-25	-25	-25	-25	-25	-25	-25	-22	-22
PT10	-19	-19	-19	-19	-19	-18	-18	-18	-19	-19	-22	-23	-23	-22	-22	-23	-25	-25	-26	-26	-25	-25	-25	-25	-22
PT 9	-19	-19	-19	-19	-19	-19	-18	-18	-19	-19	-23	-23	-23	-22	-22	-23	-25	-26	-26	-26	-26	-25	-25	-25	-26
PT 8	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-22	-23	-23	-22	-23	-23	-26	-26	-26	-26	-26	-25	-25	-25	-25
PT 7	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-23	-23	-23	-22	-22	-22	-25	-25	-26	-26	-26	-26	-25	-25	-25
PT 6	-19	-19	-19	-19	-19	-19	-19	-19	-19	-21	-22	-23	-23	-22	-23	-22	-25	-25	-25	-25	-25	-25	-25	-25	-25
PT 5	-19	-19	-19	-19	-19	-19	-19	-19	-19	-21	-21	-21	-21	-22	-22	-22	-24	-25	-25	-25	-25	-25	-25	-25	-25
PT 4	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-19	-24	-24	-25	-26	-26	-25	-25	-25	-24
PT 3	-19	-19	-19	-19	-19	-19	-18	-19	-19	-18	-18	-18	-18	-18	-19	-19	-19	-20	-24	-24	-24	-24	-24	-24	-24
PT 2	-19	-19	-19	-19	-18	-18	-18	-18	-18	-18	-18	-18	-18	-18	-18	-19	-19	-19	-20	-24	-24	-24	-24	-24	-24
PT 1	-19	-19	-18	-18	-18	-18	-19	-18	-18	-18	-18	-18	-18	-18	-18	-18	-19	-19	-19	-20	-24	-24	-24	-24	-24

COCFDINATE LEGEND
FOR
SAND STRATUM

	LATITUDE			LONGITUDE	
NODE	1	1857	PCINT	1	474
NODE	2	1917	PCINT	2	494
NODE	3	1937	PCINT	3	514
NODE	4	1957	PCINT	4	534
NODE	5	1977	PCINT	5	554
NODE	6	1997	PCINT	6	574
NCDE	7	2017	PCINT	7	594
NODE	8	2037	PCINT	8	614
NODE	9	2057	PCINT	9	634
NODE	10	2077	PCINT	10	654
NCDE	11	2097	PCINT	11	674
NODE	12	2117	PCINT	12	694
NODE	13	2137	PCINT	13	714
NODE	14	2157	PCINT	14	734
NODE	15	2177	PCINT	15	754
NODE	16	2197	PCINT	16	774
NCDE	17	2217	PCINT	17	794
NODE	18	2237	PCINT	18	814
NODE	19	2257	PCINT	19	834
NODE	20	2277	PCINT	20	854
NCDE	21	2297	PCINT	21	874
NCDE	22	2317	PCINT	22	894
NODE	23	2337	PCINT	23	914
NODE	24	2357	PCINT	24	934
NODE	25	2377	PCINT	25	954

CONTOUR GRID
FOR
SILT STRATA

	NO 1	NO 2	NO 3	NO 4	NO 5	NO 6	NO 7	NO 8	NO 9	NO10	NO11	NO12	NO13	NO14	NO15	NO16	NO17	NO18	NO19	NO20	NO21	NO22	NO23	NO24	NO25
PT25	-32	-32	-31	-31	-30	-23	-22	-22	-21	-20	-20	-20	-21	-21	-22	-23	-24	-24	-24	-24	-24	-25	-25	-25	-25
PT24	-32	-32	-32	-32	-31	-30	-23	-22	-21	-21	-20	-21	-21	-22	-23	-23	-23	-24	-24	-24	-24	-25	-25	-25	-25
PT23	-32	-32	-32	-32	-32	-31	-31	-23	-23	-22	-22	-22	-22	-23	-23	-23	-23	-24	-24	-24	-25	-25	-26	-26	
PT22	-33	-33	-33	-33	-32	-32	-31	-31	-24	-23	-23	-24	-24	-23	-23	-22	-22	-23	-23	-24	-24	-25	-25	-26	-26
PT21	-33	-33	-33	-33	-33	-32	-32	-32	-31	-24	-25	-25	-24	-24	-23	-23	-23	-23	-24	-24	-25	-25	-26	-26	-30
PT20	-33	-33	-33	-33	-33	-33	-33	-36	-36	-36	-26	-26	-25	-25	-24	-24	-24	-24	-24	-25	-25	-26	-26	-30	-30
PT19	-33	-33	-33	-33	-33	-33	-37	-37	-37	-37	-37	-36	-27	-26	-26	-26	-25	-26	-26	-26	-26	-26	-30	-30	-30
PT18	-33	-33	-33	-33	-33	-37	-38	-38	-38	-38	-38	-37	-37	-36	-36	-36	-36	-36	-36	-36	-36	-31	-31	-30	-30
PT17	-33	-33	-33	-37	-38	-38	-39	-39	-39	-39	-39	-38	-38	-37	-37	-37	-37	-37	-37	-37	-36	-31	-31	-31	-30
PT16	-33	-37	-37	-37	-38	-38	-39	-40	-40	-40	-39	-39	-38	-38	-38	-39	-39	-39	-38	-38	-37	-32	-31	-30	-30
PT15	-30	-37	-37	-37	-38	-38	-39	-39	-40	-39	-39	-39	-38	-38	-39	-39	-39	-39	-39	-38	-37	-32	-31	-30	-30
PT14	-29	-30	-30	-37	-37	-38	-38	-39	-39	-39	-39	-38	-38	-38	-38	-39	-39	-39	-39	-38	-38	-32	-31	-31	-30
PT13	-29	-29	-29	-30	-37	-37	-38	-38	-38	-38	-38	-38	-38	-38	-38	-39	-39	-39	-38	-38	-38	-37	-32	-31	-31
PT12	-28	-28	-29	-29	-30	-35	-36	-36	-36	-36	-37	-37	-38	-38	-38	-38	-38	-38	-38	-38	-38	-37	-32	-32	-31
PT11	-27	-28	-28	-29	-30	-35	-36	-36	-36	-36	-36	-37	-37	-38	-38	-38	-39	-39	-38	-38	-38	-38	-37	-32	-32
PT10	-27	-27	-28	-29	-30	-38	-38	-36	-38	-36	-36	-37	-37	-38	-38	-39	-39	-39	-39	-38	-38	-38	-37	-37	
PT 9	-27	-27	-28	-29	-38	-36	-36	-37	-37	-37	-37	-37	-37	-38	-38	-39	-39	-39	-39	-38	-38	-38	-38	-37	
PT 8	-28	-28	-28	-29	-35	-36	-36	-37	-37	-37	-37	-37	-37	-38	-38	-39	-39	-39	-39	-38	-38	-38	-38	-38	-37
PT 7	-28	-29	-29	-30	-35	-36	-36	-36	-36	-36	-36	-36	-37	-38	-38	-39	-39	-39	-39	-38	-38	-38	-38	-38	-37
PT 6	-29	-29	-29	-35	-38	-35	-36	-36	-36	-36	-36	-36	-36	-38	-38	-39	-39	-39	-39	-38	-38	-38	-38	-38	
PT 5	-29	-29	-30	-35	-35	-35	-35	-35	-36	-36	-36	-36	-36	-38	-38	-38	-38	-38	-38	-38	-38	-38	-38	-38	
PT 4	-29	-32	-32	-32	-32	-32	-32	-32	-32	-33	-33	-33	-33	-33	-37	-38	-38	-38	-38	-38	-38	-38	-39	-38	-38
PT 3	-32	-32	-32	-32	-32	-32	-32	-32	-32	-32	-32	-38	-33	-33	-33	-37	-37	-38	-38	-38	-38	-38	-37	-37	
PT 2	-32	-32	-32	-32	-32	-32	-32	-32	-32	-32	-32	-38	-38	-33	-33	-33	-33	-37	-37	-37	-37	-37	-37	-37	-37
PT 1	-38	-32	-32	-32	-32	-32	-32	-38	-32	-32	-32	-38	-38	-32	-33	-33	-33	-33	-37	-37	-37	-37	-37	-37	-37

COORDINATE LEGEND
FOR
SILT STRATUM

	LATITUDE			LONGITUDE	
NODE	1	2031	POINT	1	474
NODE	2	2045	POINT	2	494
NODE	3	2059	POINT	3	514
NODE	4	8073	POINT	4	534
NODE	5	2087	POINT	5	554
NODE	6	2101	POINT	6	574
NODE	7	2115	POINT	7	594
NODE	8	2129	POINT	8	614
NODE	9	2143	POINT	9	634
NODE	10	2157	POINT	10	654
NODE	11	2171	POINT	11	674
NODE	12	2185	POINT	12	694
NODE	13	2199	POINT	13	714
NODE	14	2213	POINT	14	734
NODE	15	2227	POINT	15	754
NODE	16	2241	POINT	16	774
NODE	17	2255	POINT	17	794
NODE	18	2269	POINT	18	814
NODE	19	2283	POINT	19	834
NODE	20	2297	POINT	20	854
NODE	21	2311	POINT	21	874
NODE	22	2325	POINT	22	894
NODE	23	2339	POINT	23	914
NODE	24	2353	POINT	24	934
NODE	25	2367	POINT	25	954

CONTOUR GRID FOR TILL STRATA

	ND 1	ND 2	ND 3	ND 4	ND 5	ND 6	ND 7	ND 8	ND 9	ND10	ND11	ND12	ND13	ND14	ND15	ND16	ND17	ND18	ND19	ND20	ND21	ND22	ND23	ND24	ND25
PT25	-54	-54	-54	-54	-54	-54	-54	-54	-54	-54	-55	-55	-55	-56	-56	-56	-56	-56	-56	-56	-56	-56	-54	-54	-54
PT24	-54	-54	-54	-54	-54	-54	-54	-54	-54	-55	-55	-55	-56	-58	-58	-58	-58	-58	-58	-58	-58	-58	-54	-54	-54
PT23	-54	-54	-54	-54	-54	-54	-54	-55	-55	-55	-58	-58	-58	-59	-59	-59	-58	-58	-58	-58	-58	-54	-54	-54	-54
PT22	-54	-54	-54	-54	-54	-54	-55	-55	-55	-58	-58	-59	-59	-59	-59	-59	-59	-59	-58	-58	-58	-54	-54	-54	-54
PT21	-54	-54	-54	-54	-55	-55	-55	-55	-56	-56	-59	-59	-59	-59	-59	-59	-59	-59	-58	-58	-58	-54	-54	-54	-54
PT20	-54	-54	-55	-55	-55	-55	-58	-58	-59	-59	-59	-59	-59	-59	-59	-59	-59	-59	-59	-58	-58	-54	-54	-54	-54
PT19	-54	-55	-55	-55	-55	-58	-58	-59	-59	-59	-59	-59	-60	-60	-60	-59	-59	-59	-59	-58	-58	-54	-54	-54	-54
PT18	-55	-55	-55	-55	-58	-58	-58	-59	-59	-59	-59	-60	-60	-60	-60	-60	-59	-59	-59	-58	-54	-54	-54	-54	-53
PT17	-55	-55	-58	-58	-58	-58	-59	-59	-59	-59	-60	-60	-60	-60	-60	-60	-59	-59	-59	-58	-54	-54	-54	-53	-53
PT16	-55	-58	-58	-58	-58	-58	-59	-59	-59	-59	-60	-60	-60	-60	-60	-60	-59	-59	-59	-58	-54	-54	-53	-53	-53
PT15	-54	-58	-58	-58	-58	-58	-59	-59	-59	-59	-60	-60	-60	-60	-60	-60	-59	-59	-59	-58	-54	-54	-53	-53	-53
PT14	-54	-54	-58	-58	-58	-58	-58	-59	-59	-59	-59	-59	-60	-60	-60	-59	-59	-59	-58	-54	-54	-54	-53	-53	-53
PT13	-54	-54	-54	-58	-58	-58	-58	-58	-59	-59	-59	-59	-59	-59	-59	-59	-59	-59	-58	-54	-54	-54	-53	-53	-53
PT12	-53	-54	-54	-54	-54	-58	-58	-58	-58	-59	-59	-59	-59	-59	-59	-59	-59	-58	-58	-54	-54	-54	-53	-53	-53
PT11	-53	-53	-53	-54	-54	-54	-58	-58	-58	-58	-58	-59	-59	-59	-59	-59	-58	-58	-58	-54	-54	-54	-54	-53	-53
PT10	-53	-53	-53	-53	-54	-54	-54	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-54	-54	-54	-54	-54	-54	-54
PT 9	-53	-53	-53	-53	-53	-53	-54	-54	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-54	-54	-54	-54	-54	-54	-54
PT 8	-53	-53	-53	-53	-53	-53	-53	-54	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58
PT 7	-53	-53	-53	-53	-53	-53	-53	-53	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58
PT 6	-52	-52	-53	-53	-53	-53	-53	-54	-54	-54	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-56	-56	-56	-58
PT 5	-52	-52	-52	-53	-53	-53	-53	-54	-54	-54	-54	-54	-54	-54	-58	-58	-58	-58	-58	-58	-58	-58	-58	-58	-59
PT 4	-52	-52	-52	-53	-53	-53	-53	-54	-54	-54	-54	-54	-54	-54	-54	-58	-58	-58	-58	-58	-58	-58	-58	-59	-59
PT 3	-52	-52	-52	-53	-53	-53	-53	-54	-54	-54	-54	-54	-54	-54	-54	-54	-58	-58	-58	-58	-58	-58	-59	-59	-59
PT 2	-52	-52	-52	-53	-53	-53	-54	-54	-54	-54	-54	-54	-54	-54	-54	-54	-58	-58	-58	-58	-58	-58	-59	-59	-59
PT 1	-52	-53	-53	-53	-53	-53	-54	-54	-54	-54	-54	-54	-54	-54	-54	-54	-58	-58	-58	-58	-58	-59	-59	-59	-59

COORDINATE LEGEND
FOR
TILL STRATUM

	LATITUDE			LONGITUDE	
NODE	1	2031	POINT	1	620
NODE	2	2040	POINT	2	630
NODE	3	2049	POINT	3	640
NODE	4	2058	POINT	4	650
NODE	5	2067	POINT	5	660
NODE	6	2076	POINT	6	670
NODE	7	2085	POINT	7	680
NODE	8	2094	POINT	8	690
NODE	9	2103	POINT	9	700
NODE	10	2112	POINT	10	710
NODE	11	2121	POINT	11	720
NODE	12	2130	POINT	12	730
NODE	13	2139	POINT	13	740
NODE	14	2148	POINT	14	750
NODE	15	2157	POINT	15	760
NODE	16	2166	POINT	16	770
NODE	17	2175	POINT	17	780
NODE	18	2184	POINT	18	790
NODE	19	2193	POINT	19	800
NODE	20	2202	POINT	20	810
NODE	21	2211	POINT	21	820
NODE	22	2220	POINT	22	830
NODE	23	2229	POINT	23	840
NODE	24	2238	POINT	24	850
NODE	25	2247	POINT	25	860

	ND 1	ND 2	ND 3	ND 4	ND 5	ND 6	ND 7	ND 8	ND 9	ND10	ND11	ND12	ND13	ND14	ND15	ND16	ND17	ND18	ND19	ND20	ND21	ND22	ND23	ND24	ND25
PT25	2	1	1	1	1	1	1	1	2	5	6	6	6	6	6	5	5	3	3	3	2	2	2	2	2
PT24	2	2	1	1	1	1	1	1	2	5	6	6	6	6	6	5	3	3	3	3	3	2	2	2	2
PT23	4	2	2	1	1	1	1	2	5	5	6	6	6	6	5	3	3	3	3	3	3	2	2	2	1
PT22	6	4	4	2	2	2	2	2	5	5	6	6	6	6	4	3	3	3	3	3	2	2	2	1	1
PT21	5	5	5	4	4	2	2	2	5	5	5	5	5	3	3	3	3	3	3	3	2	2	2	1	1
PT20	5	5	5	5	5	4	2	2	5	5	5	5	3	3	3	3	3	3	3	3	2	2	1	1	1
PT19	5	5	5	5	5	5	5	4	5	5	5	3	3	3	3	3	3	3	3	2	2	2	1	1	1
PT18	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	3	3	3	2	2	1	1	1	
PT17	5	5	5	5	5	5	5	5	4	3	3	3	3	3	3	3	3	3	3	2	2	2	1	1	1
PT16	5	5	5	5	5	5	5	5	4	3	3	3	2	3	3	3	3	3	3	3	2	2	1	1	1
PT15	5	5	5	5	5	5	5	4	4	3	3	3	2	3	3	3	3	3	3	3	2	2	1	1	1
PT14	5	5	5	5	5	5	5	4	4	3	3	2	2	3	3	3	3	3	3	2	2	2	1	1	1
PT13	5	5	5	5	5	5	4	4	4	4	3	2	2	3	3	3	3	2	2	2	2	1	1	1	1
PT12	5	5	5	5	5	4	4	4	4	3	0	0	0	0	0	0	0	0	0	2	1	1	1	1	1
PT11	5	5	5	4	4	4	4	4	4	3	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
PT10	4	4	4	4	4	4	4	4	3	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
PT 9	4	4	4	4	4	4	4	4	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
PT 8	4	4	4	4	4	4	4	4	3	3	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	1
PT 7	4	4	4	4	4	4	4	4	3	3	0	0	0	-1	-1	-1	-1	0	0	0	0	0	0	0	0
PT 6	3	4	4	4	4	4	4	4	3	3	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	0
PT 5	3	4	4	4	4	4	4	4	3	3	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	0
PT 4	3	3	4	4	4	4	4	4	3	3	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	0
PT 3	3	3	3	4	4	4	4	4	3	3	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	0
PT 2	3	3	3	3	4	4	4	3	3	3	0	0	0	0	-1	-1	-1	0	0	0	0	0	0	0	0
PT 1	3	3	3	3	4	4	4	3	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

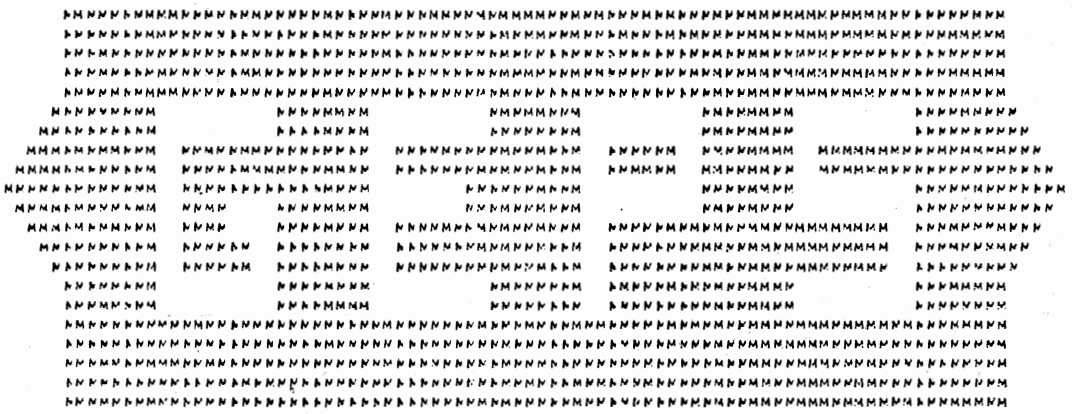
COCRDINATE LEGEND
FOR
NATA STRATUM

	LATITUDE			LCNGITUDE	
NODE	1	1771	PCINT	1	390
NODE	2	1756	PCINT	2	411
NODE	3	1821	PCINT	3	432
NCDE	4	1846	PCINT	4	453
NCDE	5	1871	PCINT	5	474
NODE	6	1856	PCINT	6	495
NODE	7	1921	PCINT	7	516
NODE	8	1946	PCINT	8	537
NCDE	9	1971	PCINT	9	558
NCDE	10	1955	PCINT	10	579
NODE	11	2021	PCINT	11	600
NODE	12	2046	PCINT	12	621
NODE	13	2071	PCINT	13	642
NODE	14	2056	PCINT	14	663
NODE	15	2121	PCINT	15	684
NODE	16	2146	PCINT	16	705
NCDE	17	2171	PCINT	17	726
NCDE	18	2156	PCINT	18	747
NODE	19	2221	PCINT	19	768
NODE	20	2246	PCINT	20	789
NODE	21	2271	PCINT	21	810
NODE	22	2256	PCINT	22	831
NODE	23	2321	PCINT	23	852
NCDE	24	2346	PCINT	24	873
NODE	25	2371	PCINT	25	894

SUCCESFUL RUN. CONGRADULATIONS - HAVE A BEER.

APPENDIX C

FIEMOD EXAMPLE LISTING



```
**********************************************
*                                            *
*                                            *
*                                            *
*            GEOTECHNICAL                    *
*                                            *
*                 ENGINEERING                *
*                                            *
*                      PLOTTING              *
*                                            *
*                          SYSTEM            *
*                                            *
*                                            *
*                 OKLAHOMA                   *
*                   STATE                    *
*                 UNIVERSITY                 *
*                                            *
*                 LEVEL II                   *
*                                            *
*                 MAY 1981                   *
*                                            *
*                                            *
**********************************************
```

```
C                                                                       00022410
C CIMMARRON VALLEY OF NORTH CENTRAL OKLAHOMA                            00022420
C                                                                       00022430
  ENGAGE FIEMOD                                                         00022440
C                                                                       00022450
C    CENTRAL                                                            00022460
C                                                                       00022470
  START                                                                 00022480
  COMMANDS    NOGRID DROP SOLID MOVE                                    00022490
  STATISTICS  SEPERATION 0.05  LINES  100                               00022500
  VIEWING ELEVATION  30.0   ANGLE  271.0                                00022510
  AREA  CNTL                                                            00022520
  IMPERIAL                                                              00022530
  POINT Z1  LATITUDE 1.0  LONGITUDE  1.0  ELEVATION  830.0              00022540
  P Z2  LA  2.0  LO 1.0  E  840.0                                       00022550
  P Z3  LA  3.0  LO 1.0  E  865.0                                       00022560
  P Z4  LA  4.0  LO 1.0  E  890.0                                       00022570
  P Z5  LA  5.0  LO 1.0  E  860.0                                       00022580
  P Z6  LA  6.0  LO 1.0  E  890.0                                       00022590
  P Z7  LA  7.0  LO 1.0  E  870.0                                       00022600
  P Z8  LA  1.0  LO 2.0  E  835.0                                       00022610
  P Z9  LA  2.0  LO 2.0  E  835.0                                       00022620
  P Z10 LA  3.0  LO 2.0  E  835.0                                       00022630
  P Z11 LA  4.0  LO 2.0  E  835.0                                       00022640
  P Z12 LA  5.0  LO 2.0  E  835.0                                       00022650
  P Z13 LA  6.0  LO 2.0  E  835.0                                       00022660
  P Z14 LA  7.0  LO 2.0  E  835.0                                       00022670
  P Z15 LA  1.0  LO 3.0  E  835.0                                       00022680
  P Z16 LA  2.0  LO 3.0  E  845.0                                       00022690
  P Z17 LA  3.0  LO 3.0  E  835.0                                       00022700
  P Z18 LA  4.0  LO 3.0  E  831.0                                       00022710
  P Z19 LA  5.0  LO 3.0  E  835.0                                       00022720
  P Z20 LA  6.0  LO 3.0  E  835.0                                       00022730
  P Z21 LA  7.0  LO 3.0  E  828.0                                       00022740
  P Z22 LA  1.0  LO 4.0  E  865.0                                       00022750
  P Z23 LA  2.0  LO 4.0  E  850.0                                       00022760
  P Z24 LA  3.0  LO 4.0  E  835.0                                       00022770
  P Z25 LA  4.0  LO 4.0  E  828.0                                       00022780
  P Z26 LA  5.0  LO 4.0  E  810.0                                       00022790
  P Z27 LA  6.0  LO 4.0  E  900.0                                       00022800
  P Z28 LA  7.0  LO 4.0  E  840.0                                       00022810
  P Z29 LA  1.0  LO 5.0  E  855.0                                       00022820
  P Z30 LA  2.0  LO 5.0  E  900.0                                       00022830
  P Z31 LA  3.0  LO 5.0  E  910.0                                       00022840
  P Z32 LA  4.0  LO 5.0  E  900.0                                       00022850
  P Z33 LA  5.0  LO 5.0  E  890.0                                       00022860
  P Z34 LA  6.0  LO 5.0  E  880.0                                       00022870
  P Z35 LA  7.0  LO 5.0  E  850.0                                       00022940
  P Z36 LA  1.0  LO 6.0  E  880.0                                       00022890
  P Z37 LA  2.0  LO 6.0  E  910.0                                       00022900
  P Z38 LA  3.0  LO 6.0  E  910.0                                       00022910
  P Z39 LA  4.0  LO 6.0  E  925.0                                       00022920
  P Z40 LA  5.0  LO 6.0  E  905.0                                       00022930
  P Z41 LA  6.0  LO 6.0  E  843.0                                       00022940
  P Z42 LA  7.0  LO 6.0  E  870.0                                       00022950
  P Z43 LA  1.0  LO 7.0  E  920.0                                       00022960
  P Z44 LA  2.0  LO 7.0  E  920.0                                       00022970
```

```
P 245 LA 3.0  LO 7.0  E  920.0                    00022980
P 246 LA 4.0  LO 7.0  E  870.0                    00022990
P 247 LA 5.0  LO 7.0  E  860.0                    00023000
P 248 LA 6.0  LO 7.0  E  870.0                    00023010
P 249 LA 7.0  LO 7.0  E  900.0                    00023020
LN'                                               00023030
```

```
* * * * END OF INPUT EXECUTION * * *
```

BORING LOCATION PLAN

BORING	LATITUDE FT/KM		LONGITUDE FT/KM		ELEVATION FEET/MTRS	
Z1	1.00/	0.3048	1.00/	0.3048	830.00/	252.9847
Z2	2.00/	0.6096	1.00/	0.3048	840.00/	256.0327
Z3	3.00/	0.9144	1.00/	0.3048	865.00/	263.6526
Z4	4.00/	1.2192	1.00/	0.3048	890.00/	271.2727
Z5	5.00/	1.5240	1.00/	0.3048	860.00/	262.1287
Z6	6.00/	1.8288	1.00/	0.3048	890.00/	271.2727
Z7	7.00/	2.1336	1.00/	0.3048	870.00/	265.1768
Z8	1.00/	0.3048	2.00/	0.6096	835.00/	254.5087
Z9	2.00/	0.6096	2.00/	0.6096	835.00/	254.5087
Z10	3.00/	0.9144	2.00/	0.6096	835.00/	254.5087
Z11	4.00/	1.2192	2.00/	0.6096	835.00/	254.5087
Z12	5.00/	1.5240	2.00/	0.6096	835.00/	254.5087
Z13	6.00/	1.8288	2.00/	0.6096	835.00/	254.5087
Z14	7.00/	2.1336	2.00/	0.6096	835.00/	254.5087
Z15	1.00/	0.3048	3.00/	0.9144	835.00/	254.5087
Z16	2.00/	0.6096	3.00/	0.9144	845.00/	257.5566
Z17	3.00/	0.9144	3.00/	0.9144	835.00/	254.5087
Z18	4.00/	1.2192	3.00/	0.9144	831.00/	253.2695
Z19	5.00/	1.5240	3.00/	0.9144	835.00/	254.5087
Z20	6.00/	1.8288	3.00/	0.9144	835.00/	254.5087

Z21	7.00/	2.1336	3.00/	0.9144	828.00/ 252.3751
Z22	1.00/	0.3048	4.00/	1.2192	865.00/ 263.6526
Z23	2.00/	0.6096	4.00/	1.2192	850.00/ 259.0806
Z24	3.00/	0.9144	4.00/	1.2192	835.00/ 254.5087
Z25	4.00/	1.2192	4.00/	1.2192	828.00/ 252.3751
Z26	5.00/	1.5240	4.00/	1.2192	830.00/ 252.9847
Z27	6.00/	1.8288	4.00/	1.2192	900.00/ 274.3206
Z28	7.00/	2.1336	4.00/	1.2192	840.00/ 256.0327
Z29	1.00/	0.3048	5.00/	1.5240	895.00/ 272.7566
Z30	2.00/	0.6096	5.00/	1.5240	900.00/ 274.3206
Z31	3.00/	0.9144	5.00/	1.5240	910.00/ 277.3687
Z32	4.00/	1.2192	5.00/	1.5240	900.00/ 274.3206
Z33	5.00/	1.5240	5.00/	1.5240	890.00/ 271.2727
Z34	6.00/	1.8288	5.00/	1.5240	880.00/ 268.2246
Z35	7.00/	2.1336	5.00/	1.5240	850.00/ 259.0806
Z36	1.00/	0.3048	6.00/	1.8288	890.00/ 268.2246
Z37	2.00/	0.6096	6.00/	1.8288	910.00/ 277.3687
Z38	3.00/	0.9144	6.00/	1.8288	910.00/ 277.3687
Z39	4.00/	1.2192	6.00/	1.8288	925.00/ 281.9407
Z40	5.00/	1.5240	6.00/	1.8288	905.00/ 275.8447
Z41	6.00/	1.8288	6.00/	1.8288	843.00/ 256.9470
Z42	7.00/	2.1336	6.00/	1.8288	870.00/ 265.1766
Z43	1.00/	0.3048	7.00/	2.1336	920.00/ 280.4167
Z44	2.00/	0.6096	7.00/	2.1336	920.00/ 280.4167
Z45	3.00/	0.9144	7.00/	2.1336	920.00/ 280.4167
Z46	4.00/	1.2192	7.00/	2.1336	870.00/ 265.1766
Z47	5.00/	1.5240	7.00/	2.1336	860.00/ 262.1287
Z48	6.00/	1.8288	7.00/	2.1336	870.00/ 265.1766
Z49	7.00/	2.1336	7.00/	2.1336	900.00/ 274.3206

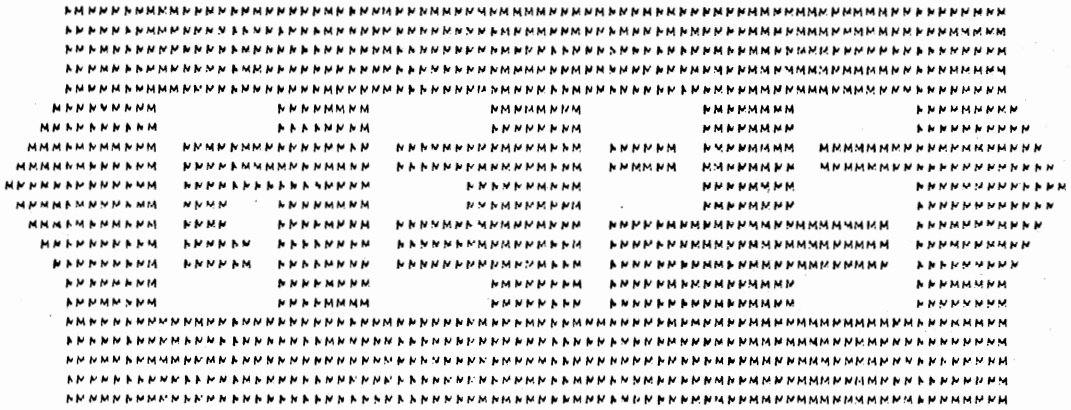
```
**********************************
*                                *
*                                *
*   THREE DIMENSIONAL PLOT       *
*                                *
*          CNTL                  *
*                                *
*                                *
**********************************
```

```
BOUNDRIES (IN INPUT UNITS)
     EAST  =              1.000000
     WEST  =              7.000000
     SOUTH =              1.000000
     NORTH =              7.000000

     SEPARATION OF THE POINTS = 0.05 INCHES
     NUMBER OF LINES USED TO FORM PLOT =  100.
     SITE VIEWING ANGLE     271.00
     ANGLE OF ELEVATION OF VIEW =     30.00
     MAXIMUM DISPLACEMENT OF PLOTTER PEN IN UPWARD DIRECTION =     10.75 INCHES
     MAXIMUM IN DOWNWARD DIRECTION =      0.0  INCHES
```

APPENDIX D

FIEMOD ERROR LISTING



```
**************************************************
*                                                *
*                                                *
*                                                *
*            GEOTECHNICAL                        *
*                                                *
*                 ENGINEERING                    *
*                                                *
*                      PLOTTING                  *
*                                                *
*                          SYSTEM                *
*                                                *
*                                                *
*                 OKLAHOMA                       *
*                   STATE                        *
*                 UNIVERSITY                     *
*                                                *
*                 LEVEL II                       *
*                                                *
*                 MAY 1981                       *
*                                                *
*                                                *
**************************************************
```

```
  ENGAGE FIEMOD                                                          00022420
C                                                                        00022430
C     CENTRAL                                                            00022440
C                                                                        00022450
  START                                                                  00022460
  COMMANDS     NOGRID DRCP SOLID MCVE                                    00022470
  STATISTICS  SEPERATION 0.05  LINES  100                                00022480
  VIEWING ELEVATION  30.0    ANGLE  271.0                                00022490
  AREA  CNTL                                                             00022500
  IMPERIAL                                                               00022510
  POINT 1  LATITUDE 1.0  LONGITUDE  1.0  ELEVATION  830.0                00022520
  P 2  LA  2.0  LO 1.0  E  840.0                                         00022530
  P 3  LA  3.0  LO 1.0  E  865.0                                         00022540
  P 4  LA  4.0  LO 1.0  E  890.0                                         00022550
  P 5  LA  5.0  LO 1.0  E  860.0                                         00022560
  P 6  LA  6.0  LO 1.0  E  890.0                                         00022570
  P 7  LA  7.0  LO 1.0  E  870.0                                         00022580
  P 8  LA  1.0  LO 2.0  E  835.0                                         00022590
  P 9  LA  2.0  LO 2.0  E  835.0                                         00022600
  P 10  LA 3.0  LO 2.0  E  835.0                                         00022610
  P 11  LA 4.0  LO 2.0  E  835.0                                         00022620
  P 12  LA 5.0  LO 2.0  E  835.0                                         00022630
  P 13  LA 6.0  LO 2.0  E  835.0                                         00022640
  P 14  LA 7.0  LO 2.0  E  835.0                                         00022650
  P 15  LA 1.0  LO 3.0  E  835.0                                         00022660
  P 16  LA 2.0  LO 3.0  E  845.0                                         00022670
  P 17  LA 3.0  LO 3.0  E  835.0                                         00022680
  P 18  LA 4.0  LO 3.0  E  831.0                                         00022690
  P 19  LA 5.0  LO 3.0  E  835.0                                         00022700
  P 20  LA 6.0  LO 3.0  E  835.0                                         00022710
  P 21  LA 7.0  LO 3.0  E  828.0                                         00022720
  P 22  LA 1.0  LO 4.0  E  865.0                                         00022730
  P 23  LA 2.0  LO 4.0  E  850.0                                         00022740
  P 24  LA 3.0  LO 4.0  E  835.0                                         00022750
  P 25  LA 4.0  LO 4.0  E  828.0                                         00022760
  P 26  LA 5.0  LO 4.0  E  830.0                                         00022770
  P 27  LA 6.0  LO 4.0  E  900.0                                         00022780
  P 28  LA 7.0  LO 4.0  E  840.0                                         00022790
  P 29  LA 1.0  LO 5.0  E  895.0                                         00022800
  P 30  LA 2.0  LO 5.0  E  900.0                                         00022810
  P 31  LA 3.0  LO 5.0  E  910.0                                         00022820
  P 32  LA 4.0  LO 5.0  E  900.0                                         00022830
  P 33  LA 5.0  LO 5.0  E  890.0                                         00022840
  P 34  LA 6.0  LO 5.0  E  880.0                                         00022850
  P 35  LA 7.0  LO 5.0  E  850.0                                         00022860
  P 36  LA 1.0  LO 6.0  E  880.0                                         00022870
  P 37  LA 2.0  LO 6.0  E  910.0                                         00022880
  P 38  LA 3.0  LO 6.0  E  910.0                                         00022890
  P 39  LA 4.0  LO 6.0  E  925.0                                         00022900
  P 40  LA 5.0  LO 6.0  E  905.0                                         00022910
  P 41  LA 6.0  LO 6.0  E  843.0                                         00022920
  P 42  LA 7.0  LO 6.0  E  870.0                                         00022930
  P 43  LA 1.0  LO 7.0  E  920.0                                         00022940
  P 44  LA 2.0  LO 7.0  E  920.0                                         00022950
  P 45  LA 3.0  LO 7.0  E  920.0                                         00022960
  P 46  LA 4.0  LO 7.0  E  870.0                                         00022970
  P 47  LA 5.0  LO 7.0  E  860.0                                         00022980
  P 48  LA 6.0  LO 7.0  E  870.0                                         00022990
```

ERROR 4

EXPECTING A REAL VALUE FOR LATITUDE

VITA

Mark Wallace Pabst

Candidate for the Degree of

Master of Science

Thesis: COMPUTER AIDED REDUCTION OF GEOTECHNICAL ENGINEERING BORING DATA

Major Field: Civil Engineering

Biographical:

Personal Data: Born in Beaver Dam, Wisconsin, November 24, 1955, the son of Mr. and Mrs. J. W. Pabst.

Education: Graduated from Putnam City West, Oklahoma City, Oklahoma, in May, 1974; received Bachelor of Science in Civil Engineering degree from Oklahoma State University in July, 1978; entered graduate school, Oklahoma State University in June, 1980, and completed requirements for the Master of Science degree in May, 1981.

Professional Experience: Alumnus of Triangle Fraternity. Employed by Mueser, Rutledge, Johnston, and DeSimone in New York, New York from August, 1978 to April, 1980 as an entry level Soils Engineer.