An Interpretive Finite Element Problem Oriented Language For Hydraulic Engineering

A Thesis
Presented for the Degree of Doctor of Philosophy of the University of Southampton in the Faculty of Engineering and Applied Science

by

JAMES MATTHEW NELSON

September 1976
This thesis describes the design and implementation of a major finite element computer system. The system recognizes a problem oriented language called FEH POL (Finite Element Hydrodynamic Problem Oriented Language). Its function is to allow the user to initiate and coordinate the solutions of given partial differential equations on two dimensional regions. By issuing some simple commands, models can be developed, modified, and run with complete interactive user control at each phase.

Among the system's facilities which are provided are symbolic representation and manipulation of geometrical entities (such as curves, regions, and nodes,) and solution parameters (state variables, boundary conditions, constants). An original mesh generation algorithm for multinoded triangular elements is included, together with comprehensive display and plotting capabilities. The system processes eight supervisor calls, mostly for passing finite element data to and from the solution program(s).

Several examples of solved problems are given, and techniques for programming in FEH POL are discussed. In addition, a short discussion of the existing solution programs is included.
<table>
<thead>
<tr>
<th>CONTENTS</th>
<th>page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>2</td>
</tr>
<tr>
<td>List of terms and abbreviations</td>
<td>4</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td></td>
</tr>
<tr>
<td><strong>Chapter 1:</strong></td>
<td></td>
</tr>
<tr>
<td>(1.1) Finite Element Systems</td>
<td>5</td>
</tr>
<tr>
<td>(1.2) The FEHPOL System</td>
<td>8</td>
</tr>
<tr>
<td>(1.3) The FEHPOL Language</td>
<td>8</td>
</tr>
<tr>
<td><strong>Chapter 2:</strong></td>
<td></td>
</tr>
<tr>
<td>(2.1) Parsing Scheme</td>
<td>11</td>
</tr>
<tr>
<td>(2.2) Error Detection and Recovery</td>
<td>14</td>
</tr>
<tr>
<td>(2.3) Documentation</td>
<td>16</td>
</tr>
<tr>
<td><strong>Chapter 3:</strong></td>
<td></td>
</tr>
<tr>
<td>(3.1) FEHPOL Programs</td>
<td>20</td>
</tr>
<tr>
<td>(3.2) Defined Types</td>
<td>21</td>
</tr>
<tr>
<td>(3.3) Supervisor Calls</td>
<td>23</td>
</tr>
<tr>
<td>(3.4) The Symbol Pool</td>
<td>26</td>
</tr>
<tr>
<td>(3.5) Machine Independence</td>
<td>28</td>
</tr>
<tr>
<td><strong>Chapter 4:</strong></td>
<td></td>
</tr>
<tr>
<td>(4.1) Some FEHPOL Algorithms</td>
<td>30</td>
</tr>
<tr>
<td>(4.2) The Triangular Mesh Generation Algorithm</td>
<td>30</td>
</tr>
<tr>
<td>(4.3) Algorithm to Generate Side Nodes</td>
<td>36</td>
</tr>
<tr>
<td>(4.4) Internal Node Generation Algorithm</td>
<td>36</td>
</tr>
<tr>
<td>(4.5) Contour, Plotting Algorithm</td>
<td>40</td>
</tr>
<tr>
<td>(4.6) Smooth Interpolation</td>
<td>45</td>
</tr>
<tr>
<td><strong>Chapter 5:</strong></td>
<td></td>
</tr>
<tr>
<td>(5.1) FEHPOL Solution Programs</td>
<td>56</td>
</tr>
<tr>
<td>(5.2) The Solution Program DSPSTD (examples)</td>
<td>57</td>
</tr>
<tr>
<td>(5.3) The Solution Program NAVSTK (example)</td>
<td>75</td>
</tr>
<tr>
<td>(5.4) General Examples of Use of FEHPOL</td>
<td>85</td>
</tr>
<tr>
<td><strong>Chapter 6:</strong></td>
<td></td>
</tr>
<tr>
<td>(6.1) Conclusions and Discussion</td>
<td>107</td>
</tr>
<tr>
<td>(6.2) Enhancements and Further Development</td>
<td>108</td>
</tr>
<tr>
<td>Appendix</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>A</td>
<td>The FEHPOL System Source</td>
</tr>
<tr>
<td>B</td>
<td>The FEHPOL User's Manual</td>
</tr>
<tr>
<td>C</td>
<td>The Solution Program NAVSTK</td>
</tr>
<tr>
<td>D</td>
<td>Adding Solution Programs to FEHPOL</td>
</tr>
<tr>
<td>E</td>
<td>Expanded Introduction and Discussion of Previous Finite Element Systems</td>
</tr>
</tbody>
</table>
Acknowledgements

I would like to express my thanks to Dr. C. A. Brebbia for his suggestions and direction through each phase of the project. Thanks are also due to Dr. R. A. Adey, for his frequent comments and many hours of discussion. Thanks also to him for providing a useful finite element modelling program to be used in conjunction with the FEHPOL System.

To Dr. P. Partridge, C. Roach, Ms. J. Gosink, and S. Smith, many thanks for suggestions relating to the syntax of the FEHPOL language.

Finally, thanks to each of the participants at the December 1975 Introductory Course on Finite Elements for Fluid Flow, both for acting as 'guinea pigs' and for suggestions resulting in several desirable modifications and additions to the system.
List of Terms and Abbreviations

bandwidth - the maximum number of elements in any row of a matrix outside which every matrix element is equal to zero.

BNF - (Backus-Naur Form) a language for describing the grammar of any language.

connectivity - a table of the node numbers which are associated with each element in the finite element mesh.

finite state automaton - (deterministic) a five-tuple \((K, V_T, M, S, Z)\) where \(K\) is a set of states, \(V_T\) is the input alphabet, \(M\) is a mapping from \(K \times V_T\) into \(K\), \(S\) is the start state, and \(Z\) is the nonempty set of final states.

grammar - (denoted \(G(Z)\)) a nonempty set of production rules with distinguished symbol \(Z\).

interpreter - a program recognizing a language which performs the requested operations itself.

nonterminal symbols - symbols that appear in a grammar, but not in the language it defines.

parsing - recognizing a language.

production rule - a rule for replacing a set of symbols by a single nonterminal symbol.

side nodes - nodes not situated on the vertex of a finite element.

state diagram - a pictorial representation of a finite state automaton.

state transition - proceeding from one state to the next via the mapping \(M(Q,T)\) for a given FSA.

supervisor call - a request to a supervisory system to
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>terminal symbols</td>
<td>symbols appearing in the language defined by a given grammar.</td>
</tr>
<tr>
<td>top-down parsing</td>
<td>a predictive recognition process that avoids use of precedence relations to form the parse.</td>
</tr>
</tbody>
</table>
Introduction

The aim of this thesis is to provide a detailed description of the design, implementation, and use of the FEHPOL System. It begins with a discussion of the aims and requirements of an interactive finite element system and of how the FEHPOL System has met them.

Chapter Two discusses the actual development strategy, including a discussion of the parsing scheme. Chapter Three goes into some detail about the structure of the FEHPOL System. The supervisor calls and symbol structure are described.

The fourth chapter deals with several of the original algorithms contained in the system which have general utility.

Chapter Five presents a variety of steady state dispersion, seepage, and flow problems and their solution using the FEHPOL system. Examples of FEHPOL generated graphics and listings of sequences of FEHPOL commands are given.

The sixth chapter consists of concluding remarks and some suggestions for the further development of the system.
CHAPTER 1

DISCUSSION
1.1 Finite Element Systems

While the finite element method is widely used in structural mechanics problems, its relatively recent extension to problems in fluid dynamics has become increasingly practicable. Some reliable codes are in production use which model seepage, thermal problems, estuarial hydrodynamics, and related problems. Several more codes are under development for such problems as wave mechanics, turbulent flow, and fluid-structure interactions.

The prime difficulties in using these codes lie in the batch input procedure. Because the data is generally complex and interrelated, problem descriptions are error prone. The rigidity of the data does not allow convenient modification of the problem and the formation of a finite element idealization which forms an adequate approximation to the problem but also has optimal computability is rather difficult.

The advantages of computer aided data preparation are readily evident, and thus the proliferation of preprocessing programs has met the demands of the non-design oriented sector. Several large finite element systems have been developed, particularly for use in static stress analysis. These combine a wide variety of solution capabilities with both pre and post-processing, and have met with substantial success among the engineering community.

With the rapid development of distributed processing and wide range of powerful minicomputers becoming available, smaller interactive computer systems are being developed. These often suit the classes of problems which are either uneconomic on the large scale systems or which are primarily design oriented, where problem descriptions are the independent variable and immediate feedback is essential.

While interactive computer aided design systems have been de-
veloped for use with the finite element technique, these primarily are data generation systems which use interactive graphics to help the designer define and modify his problem description. Particularly for the developing area of coastal and offshore engineering, a computer system is needed which is able to interactively coordinate the data generation, solution, and subsequent analysis of the design problem.

Some requirements of such a system may be summarized as follows:

1) It must be easy to use.
2) Its response time must be short enough for the designer to maintain continuity of thought.
3) It must have low core storage requirements to allow use on modern minicomputers.
4) The commands must allow minimum problem description without loss of versatility.
5) It should be able to modify designs without having to repeat the whole process from the beginning.
6) It should diagnose any errors in an easily understandable way.
7) It should be crash proof.

Characteristic of the development of computer systems for any scientific or engineering purpose is the conflict between the systems programmer or software specialist and the engineer or scientist. Only by a well balanced partnership between the two groups can an efficient and modular, yet user and problem oriented system be developed.
1.2 The FEHPOL System

To effectively use the existing solution codes and the engineering expertise they represent to maximum benefit, the system was designed and structured in many ways similar to an ordinary operating system. Rather than incorporating solution codes within the basic structure of the system, they are run under the system in a general and installation independent way. The command language is acting not only as an interactive data generation and display initiator but also as the job control language for executing finite element solution programs of arbitrary capabilities. Commands are available to assign and reassign system devices as in an ordinary job control language. In fact, coupled with the local device service routines for a particular installation, the FEHPOL system may run as a stand-alone non-multitasking operating system.

This design strategy separates the system's software strategy from the actual application programs, thus providing the necessary balance between systems programming techniques and engineering expertise. In addition, it allows a very open ended development of codes to run under the system for a wide variety of problem types, while making use of existing reliable codes with a minimum of alteration. As the system is an interpreter, no precompiling or object code generation is required. The system simply reads its own problem oriented language and executes the intended operation.

1.3 The FEHPOL Language

In developing the syntax of the FEHPOL language much care and attention was given to make it as easy to learn and as similar to English as possible. Feedback from consulting engineers, academic staff, and conference delegates helped to crystallize the development of a workable and understandable set of commands. Not only
does a problem oriented language of this nature need to be able to accept minimum programmer description of a finite element problem, but also it must be versatile enough to represent a wide range of problem types.

To accomplish this, statements should be independent of one another and mutually transparent. They must perform actions on the data structure which leave its integrity unimpaired regardless of the possibly illegal consequences of the action. Being a high level language (one in which a single instruction may represent very complex operations) rather than basing it on primitives, the number of different command types is minimized and consequently it is easier to learn. The aim is to provide a wide range of modifiers to the verbs to enable logically similar but computationally different operations to be performed. An example of this is the PLOT command. In each of the following cases a different type of plot is generated:

- **PLOT NODES**
- **PLOT MESH**
- **PLOT AND LABEL MESH**
- **PLOT <name of nodes>**
- **PLOT <name of region>**
- **PLOT <name of curve>**
- **PLOT CONTOUR MAP OF <name of parameter>, LEVELS=(<list of numbers>)**

To help make the language easy to learn and use, the lexical analysis is very flexible. Free formatting is supported, and the ability to recognize abbreviations of keywords helps simplify use and eliminate spelling errors. Both blank spaces and blank lines are allowable anywhere but within a name or keyword.
CHAPTER 2

DEVELOPMENT STRATEGY
2.1 Parsing Scheme

Formal language theory has been developed considerably since Chomsky first described a language formally. The definition of a grammar which generates a language and the finite state automaton have been of great benefit to compiler writers, not so much in terms of the relative merit of formal parsing techniques over ad hoc schemes, but by providing considerable insight into the recognition process. A basic knowledge of automatic syntax methods clearly helps him to program more systematically and efficiently.

A state diagram is a representation of a finite state automaton. It is more closely suited to aiding the compiler writer than the more rigorous mathematical representation. By allowing nonterminal symbols of the language in the state diagram one can effectively nest diagrams in an easily understandable form. For instance, consider a small subset of the production rules for the FEHPOl language (written in BNF):

\[
\begin{align*}
\text{<curvestatement> ::= } & \text{CURVE} \text{<pointlist><identifier>} \\
\text{<pointlist> ::= } & \text{<point>} \\
\text{<pointlist> ::= } & \text{<pointlist><point>}
\end{align*}
\]

In state diagram form these could be represented by Figure 1.
Starting at the start state (labelled S) of a given diagram and traversing arcs in clockwise preference whenever the symbol above it is recognized, the end state (labelled E) is reached if the input sentence was syntactically correct. If the end state cannot be reached, a syntax error is detected.

To include both the actions which correspond to each state transition and proper error analysis and recovery, more features can be added to the diagramming representation. Differentiation must be made between errors which occur within the state diagram and those which occur because the first symbol in the diagram is not recognized. This requires three standard exits to each diagram which we may denote EX700 (exit for errors occurring within a diagram), EX800 (normal exit without error), and EX900 (recognition did not begin). Secondly, the actions associated with a state transition may be included below the arc and denoted ACTn. This means that the action must be performed when traversing the arc. Using these conventions the diagram on the preceding page becomes Figure 2.

![Figure 2](image-url)
In this form, the diagram has become essentially a flowchart of the interpretation process, which may be coded quite easily in FORTRAN. Let each diagram represent a FORTRAN function which returns one of three possible values, depending on the exit taken. For convenience the values are chosen to be (-1,0,1) which correspond to the exits EX700, EX800, and EX900. With the aid of the lowlevel routine NEXTIS, which reads cards and scans for the next incoming character string, returning the value zero whenever the string matches its argument, the transition diagram may be coded as follows:

```fortran
FUNCTION CURVESTATEMENT(3)
  IF(NEXTIS('CURVE'))700,10,900
  10 ...perform action 1...
  IF(POINTLIST(3))700,20,700
  20 ...perform action 2...
  IF(IDENTIFIER(3))700,30,700
  30 ...perform action 3...
  GO TO 800
700 CURVESTATEMENT--1
  RETURN
800 CURVESTATEMENT--3
  RETURN
900 CURVESTATEMENT--1
  RETURN
END
```

In this way the interpreter is coded in a compact, systematic and efficient manner. The program is effectively modularized in such a way that the grammar is readily apparent and easily alterable.
2.2 Error Detection and Recovery

The systematic interpretation scheme discussed in the previous section makes the recognition of syntax errors an automatic process. The system replies to a syntactical error by printing the offending line with a pointer to the character at which the error occurred and waiting for the next command. If the user is not sure of the correct syntax of the command, he may trace the parsing process by preceding it with the command:

```
TRACE ON
```

the effect of this statement is to provide a map of the parsing of subsequent statements. An example of the use of this command is given in Example 1 on the next page. The map lists the character being sought by NEXTIS, as well as the name of the routine representing the state diagram just entered. Because the names (IDTs) of the routines are picked so that the last four letters abbreviate the name of the nonterminal symbol being sought, the state diagram paths taken are clearly evident.

In the example, the parser first looked for a CURVE statement (FPCUST). To find one it looked for the character 'C'. As it was not found the parser went on to search for an INTERPOLATE statement (FPINST) which it found. Next it looked for a defined type name (FPDINM) which required looking for letters, the character '-', and digits (FPLETR and FPDIGT). Following this sort of map one can easily see not only that the comma was the offending character, but also the other possible characters that could take its place and be syntactically correct. By this means, the user can figure out the syntax of the commands online, without resorting to a manual every time he forgets the exact format.
Example 1
Besides making syntax errors, the user may make logical errors, such as referencing an undefined symbol or attempting an illegal operation. Errors of this nature are handled on a case by case basis with careful checks. In the case of most logical errors, a concise, self-explanatory message is printed and the remainder of the current line is ignored. However, for certain errors, the system actually diagnoses and corrects the fault and continues processing the current line. In Example 2 on the next page are given several examples of logical error diagnosis and recovery.

A third type of FEHPOL errors have to do with the limits of the system and the integrity of the data structure. Among these are symbol pool overflow, overflow of workareas, corruption of various tables, and solution program error exits. They cause a message to be displayed of the form:

```
***ABORT, CODE= nnn****
```

where nnn is the ABORT code (given in Table 1). After an abort the system reinitializes itself and processing is resumed.

2.3 Documentation

Good documentation is essential if a computer system is to be of general applicability and wide use. Unfortunately, it is in this area that most engineering systems, and a great deal of powerful general purpose software are lacking. The documentation strategy for the FEHPOL system is twofold. Firstly, the manuals for user's and programmers are separate, accurate, and as up-to-date as possible, and secondly, comprehensive documentation is included in comment form as part of the system's source.
QUIT
WRITE CONNECTIVITY
*CONNECTIVITY TABLE IS EMPTY*
WRITE CONNECTIVITY

CURVE 1,1 2A6.2:JWON-1
*SYNTAX ERROR!
CURVE 1,1 2A6.2:JWON-1

:A
*SYNTAX ERROR!

INTERPOLATE S NODES=10,LINEAR
*UNDEFINED NAME!
INTERPOLATE S NODES=10,LINEAR

SET Q=3.3 ON D
*UNDEFINED NAME!
SET Q=3.3 ON D

REGION 1,1 1,2:CRUMMY-REGION
MESH CRUMMY-REGION WITH THREE NODED TRIANGLES
*TOO FEW BOUNDARY NODES!
MESH CRUMMY-REGION WITH THREE NODED TRIANGLES

GENERATE INTERNAL NODES ON NONEXISTENT
*UNDEFINED NAME!
GENERATE INTERNAL NODES ON NONEXISTENT

REGION R 1,1 1,4 4,4 4,1:A
*UNDEFINED NAME!
REGION R 1,1 1,4 4,4 4,1:A

NODE 3,3:4,4:A
*NAME NOT UNIQUE!
NODE 3,3:4,4:A

NAME NOT UNIQUE!! CHANGED TO X
REGION CRUMMY-REGION:B
MESH A HOLES=(B) WITH SIX NODED TRIANGLES
*TOO FEW NODES ON INTERNAL BOUNDARY!!
MESH A HOLES=(B) WITH SIX NODED TRIANGLES

LET -7,-7=INTERNAL NODE FOR A
MESH A WITH THREE NODED TRIANGLES
*UNUSUAL MESH OPTIMIZATION OMITTED!
MESH A WITH THREE NODED TRIANGLES

SET B=13 ON #123
*NODE NUMBER TOO LARGE!
SET B=13 ON #123

Example 2
TABLE OF ABORT CODES FOR THE FEHPOL SYSTEM 15/08/76

<table>
<thead>
<tr>
<th>ABORT NO.</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-28</td>
<td>RESERVED FOR SOLUTION PROGRAMS</td>
</tr>
<tr>
<td>29</td>
<td>SYMBOL OR PARAMETER POOL OVERFLOW</td>
</tr>
<tr>
<td>30</td>
<td>PARAMETER POOL CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>31</td>
<td>SYMBOL POOL CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>32</td>
<td>MORE THAN 100 SYNTAX ERRORS SINCE INIT/QUIT</td>
</tr>
<tr>
<td>33</td>
<td>ILLEGAL INTERPOLATION CODE</td>
</tr>
<tr>
<td>34</td>
<td>ILLEGAL ELEMENT CODE</td>
</tr>
<tr>
<td>35</td>
<td>NEGATIVE NUMBER OF INTERNAL NODES</td>
</tr>
<tr>
<td>36</td>
<td>NO. OF EQUALLY ATTRACTIVE CONNECTIONS &gt;10</td>
</tr>
<tr>
<td>37</td>
<td>TOO MANY FINITE ELEMENT NODES FOR SYSTEM SIZE</td>
</tr>
<tr>
<td>38</td>
<td>RENUMBERING TABLE CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>39</td>
<td>TOO MANY FINITE ELEMENT CORNER NODES</td>
</tr>
<tr>
<td>40-45</td>
<td>RESERVED FOR SYSTEM DEVELOPMENT</td>
</tr>
<tr>
<td>46</td>
<td>NOT ENOUGH POOL SPACE LEFT FOR INTERPOLATE</td>
</tr>
<tr>
<td>47-52</td>
<td>RESERVED FOR SYSTEM DEVELOPMENT</td>
</tr>
<tr>
<td>53</td>
<td>ATTEMPTED CONTOUR PLOT WITH UNAVAILABLE ELEMENT</td>
</tr>
<tr>
<td>54</td>
<td>ATTEMPTED CONTOUR PLOT OF NON-NODAL PARAMETER</td>
</tr>
</tbody>
</table>

NOTE: A MORE DETAILED DESCRIPTION OF THE MEANING OF THESE CODES AND CORRECTIVE ACTION TO BE TAKEN SHOULD THEY OCCUR IS GIVEN IN THE USER'S MANUAL ENTITLED 'THE FEHPOL SYSTEM'.

Table 1
CHAPTER 3

THE SYSTEM'S STRUCTURE
3.1 FEHPOL PROGRAMS

FEHPOL is a special purpose interpretive problem oriented language for setting up and solving hydraulic engineering problems using the finite element method. Its prime function is to provide a user oriented interface between the complexities of finite element calculations and the description of a particular problem. This is accomplished using a command processing supervisor structure.

Among the facilities which the FEHPOL system provides are symbolic definition, reference, and manipulation of geometrical entities (such as curves, regions, and nodes) and of solution parameters (such as state variables, boundary conditions, etc), automatic node generation of several types, automatic mesh generation and bandwidth minimization, comprehensive display and plotting facilities, physical device reassignment, an internal trace facility, and the ability to load and execute arbitrary solution programs.

The language is specifically tailored to solving partial differential equations on two dimensional geographically oriented regions. As such, the commands are of a very high level in terms of the complexity of the response to a given command. The symbolically referenceable data types are lists of points and lists of data associated with a solution program (parameters).

Being an interpreter, the system builds its tables dynamically using a two way pool for maximum storage efficiency. The parsing algorithm used is top-down to facilitate addition of commands and to eliminate the dependence on a strict set of precedence relations. The efficiency lost by this technique is made up by ease of maintenance and alteration. The system includes a command frequency table to speed parser operation.
3.2 Defined Types

A CURVE in FEHPOL is an ordered list of two or more points which represents a non-closed curve. It is associated with a symbolic name when defined, and can be referenced in many different FEHPOL commands by name. For instance, a CURVE may be operated upon by the INTERPOLATE statement producing a new list of points, larger than the original list, according to one of several user specifiable interpolation formulae.

Reference can be made to a particular point in a CURVE by using subscripts (similar to FORTRAN arrays). For instance, the tenth point in CURVE named ISLEOFWIGHT could be referred to as ISLEOFWIGHT(10). In addition, parts of CURVES may be referenced using expanded subscripted format (i.e. SEAWALL(3-11)). A CURVE may also be referenced in reverse order by preceding it with a minus sign.

CURVES are the fundamental unit in building a description of the geometry of a region on which a finite element mesh is to be produced and a partial differential equation solved. The various interpolation types allow minimum description of region boundaries yet with an arbitrary number of finite element nodes on them. In particular, the interpolation types REGULAR and SMOOTH provide a simple way to define the typically unsymmetric boundaries of seas, harbours, and estuaries.

Because nodal solution parameters such as boundary conditions are specified on CURVES, CURVES generally represent the logical parts of a region's boundaries. For instance, to solve a differential equation on a square box with the state variable PSI equal to -13.0 on one side of the region and 5.0 on another side one could define four CURVES, named RIGHTSIDE, BOTTOMSIDE, LEFTSIDE, and TOPSIDE and the region SQUARE in terms of them.
then he could set the value of PSI on the sides by issuing the command:

```
SET PSI=-13.0 ON HIGHSIDE, PSI=5 ON LEFTSIDE
```

If a mesh has been generated which contains side noded elements, such as six noded triangles, setting a parameter on a CURVE automatically sets the side nodes on any side which is along the CURVE.

A REGION is simply a closed curve. It is referenced like a CURVE but represents a two dimensional simply connected region. Setting a parameter on a REGION means that each finite element contained in the REGION has the value assigned, rather than any node. Several operations are allowed on REGIONS, such as generating internal nodes for a REGION, meshing a REGION with finite elements, and omitting a REGION from a finite element mesh.

Generally, a REGION is defined in terms of CURVES and NODES to represent the region upon which the differential equation(s) are to be solved, or to provide a specification of a subsection of that region on which a particular property is to be set.

NODEs are collections of points (like CURVES) but with no order implied. For instance, a list of internal nodes for a particular REGION, or a list of nodes upon which a certain boundary condition is to be set. NODEs are usually defined explicitly as an intermediate step in defining both CURVES and REGIONS. They allow easy alteration of the shape of a REGION when the REGION is defined indirectly in terms of them.

Parameters are simply named lists of numbers for use with the solution program. By issuing the appropriate su-
pervisor calls, the solution program may access any list of data which has been set by the user. In addition, parameters may be set by the solution program using the FPSPUT supervisor call. This enables solution values to be passed back to the system, where the user can examine them or use the PLOT command to draw contour maps or vector plots.

3.3 Supervisor Calls

The function of the FEHPOL system is to provide a solution program with finite element data input and output in a standardized form. This is done by allowing the solution program to issue supervisor calls to the system. These are coded as FORTRAN call statements, or in the case of FPSPMS, as a FORTRAN function call. Among the SVCs available are the following:

- **FPSTOP** return to FEHPOL system
- **FPELCY** fetch element connectivity
- **FPNCDS** fetch nodal coordinates
- **FPNPIPS** fetch mesh statistics
- **FPITITL** fetch job title
- **FPSPMS** fetch a solution parameter
- **FPQERY** fetch solution parameter statistics
- **FPSPUT** return a parameter to the system

Under FEHPOL's supervision, the solution program needs to concern itself only with assembling and solving a system of equations according to a given differential equation.

The format of a call to FPSTOP is as follows:

```
CALL FPSTOP(I)
```

where I is a code denoting the termination status. If I is zero, the FEHPOL system is returned to normally. Otherwise, an ABORT message is printed on the REPLY device giving the
code I to reference the error or illegal condition. Effectively, FPSSTOP is the FEHPOL equivalent of the FORTRAN STOP statement, which returns to the user's operating system.

The call to FPELCK is of the following form:

```plaintext
CALL FPELCK(I)
```

where ICON is an array into which the element connectivity of the Ith element is passed.

The service routine FPNCDS is called as follows:

```plaintext
CALL FPNCDS(M,N,X,Y)
```

In this case, M is the sequence number of the finite element nodes. N is returned as the optimized node number of the Mth node and X and Y are the coordinates of that node. The routine is written this way to provide convenient alteration of existing programs for running under the FEHPOL system.

FPNPMS passes to the solution program the statistics of a finite element mesh. The call is:

```plaintext
CALL FNPMS(MEL q NCP, N, LNY, LTY, IBD)
```

where:

- `MEL` = number of elements
- `NCP` = number of corner nodes (non-side nodes)
- `N` = number of nodes total
- `LTY` = maximum number of nodes per element
- `IBD` = semi-bandwidth

The call to FPITITL has the following form:

```plaintext
CALL FPITITL(A)
```

where A is an array that can contain seventytwo characters. The job title string specified in the EXECUTE statement is passed to the solution program with this call. This can be useful particularly when running the system in batch mode.
The call to FPSPMS is coded as a FORTRAN function call. An example of the call is:

\begin{verbatim}
A=FPSPMS(11,WAVE-HEIGHT,11,NUM,IFLAG)
\end{verbatim}

here, WAVE-HEIGHT is the name of a FEHPCL parameter. It may be either a global, elemental, or nodal parameter. If NUM is positive, the function returns the value of that parameter on the NUMth element or node. If num is negative, the function returns the value of WAVE-HEIGHT on the NUMth node or element on which it was actually set. Accessing the value of a parameter on an element or at a node on which it was not set causes the value to be returned undefined (-1111.0). In addition, IFLAG is returned nonzero in this case or in the event of any irregular condition. The meaning of the values of IFLAG are as follows:

<table>
<thead>
<tr>
<th>IFLAG</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>the parameter was never defined</td>
</tr>
<tr>
<td>2</td>
<td>NUM is too negative (not that many nodes or elements had the value set)</td>
</tr>
<tr>
<td>3</td>
<td>NUM is too positive (greater than the length of the list)</td>
</tr>
<tr>
<td>4</td>
<td>the parameter is not defined at the specified node or element</td>
</tr>
</tbody>
</table>

FPQERY is similar in form to FPSPMS. It is called in the following manner:

\begin{verbatim}
CALL FPQERY(10,PARAM-NAME,10,N,M)
\end{verbatim}

In this supervisor call, if N equals -1 the number of nodes or elements at which the parameter named PARAM-NAME was set is returned in M. If N is positive, the node number (or element number) of the Nth set value of the parameter is returned in M. If the parameter was not defined, M is returned equal to zero. This is used for finding parameter statistics before accessing them.
For instance, the loop in a solution program to consider the nodal boundary condition \texttt{U-VEL} might look like:

\begin{verbatim}
CALL FPQER (SHD-VEL,5,-1,MUBEC)
DO 10 I=1,MUBEC
UVELEC(I)=FPFSUB(SHD-VEL,5,-1,JP)
10 CONTINUE
\end{verbatim}

The \texttt{FPSPUT} supervisor call returns computed data from the solution program to the \texttt{FEHPOP} data structure. It is coded as follows:

\begin{verbatim}
CALL FPSPUT(SHIMME,4,MIM,ARRAY)
\end{verbatim}

This defines a new \texttt{FEHPOP} parameter named \texttt{NAME} which consists of the first \texttt{MIM} elements of the \texttt{FORTRAN} array \texttt{ARRAY}. For example, to pass back the solution of a simple scalar potential problem the call might look like:

\begin{verbatim}
CALL FPSPUT(SOLUTION,8,MNMRP,PSI)
\end{verbatim}

where \texttt{PSI} is the solution vector. By issuing a \texttt{PILOT} command to the system which referenced this parameter a contour plot of the solution could then be produced.

3.4 The Symbol Pool

The single most important design feature of the system is the symbol pool. This is a two-way table which contains names of CURVES, REGIONS, NODES, and parameters, together with indices and the actual data they represent. Each definition builds each end of the table toward the other until the pool is full. In this way storage is used most efficiently and the system size is easily altered.

The basic cell in the pool structure is one \texttt{FORTRAN} real variable. The system, however, allocates space and accesses information in a way that is independent of the local hardware and \texttt{FORTRAN} compiler. The system calculates the number of characters
The Pool Structure

Figure 3
that one real variable can hold on the local computer and calculates the pool entry size requirements based on this.

3.5 Machine Independence

Several other system features require hardware and compiler independent packing and storage of character data. The assembly language routines COMP and COPY are used to make up for the lack of character manipulation in FORTRAN. With these exceptions and also the coding of the lowest level plotting routines which are device dependent, the system is written entirely in ANSI FORTRAN IV.
CHAPTER 4

FEHPOL ALGORITHMS
4.1 Some FEHPOL Algorithms

While many algorithms used in the system are tailored specifically to the system's data structure and syntactical analysis scheme, several of the algorithms have quite general stand-alone applicability. In particular, four original algorithms are of practical import.

Probably the most important of these is the triangular mesh generation algorithm, which is simpler and more efficient than other algorithms. A rather trivial, but useful algorithm which generates side noded element connectivity from three noded connectivity while preserving optimal node numbering is given. In addition, an internal node generation algorithm which uses boundary node density to determine local internal node density, thus requiring only a boundary specification is discussed. Lastly, a general purpose contour plotting algorithm which uses the shape function of the finite elements rather than rectangular interpolation to locate and plot contours is described.

4.2 The Triangular Mesh Generation Algorithm

Although several algorithms are available for generation of triangular finite element meshes on given regions, notably the modified Suhara-Fukudu algorithm and the method of isoparametric coordinates, the necessity of representing the typically irregularly shaped regions associated with coastal and estuarial problems and the demand for fast interactive response time required a more flexible and less time-consuming algorithm. The method of isoparametric coordinates, although quite useful when generating a mesh on a regularly shaped region, becomes less attractive when applied to practical problems involving irregular boundaries (for instance, Example 11), because of the amount of user specified zoning which must take place.
The use of the modified Suhara-Fukudu algorithm was decided against based on the author's opinion that it is unnecessarily complicated and time consuming. The algorithm requires a large amount of checking to see if a potential element overlaps a previously generated element or boundary side. By choosing a better suitability criterion, the overlap check may be avoided completely, and the boundary check limited to the most suitable of potential elements. In addition, the use of a push-down stack to save the unconnected triangles' sides in most likely order seems more efficient than picking a side from previous elements.

In choosing a suitability criterion for triangular elements one may consider the following problem:

Given any two of several points as a baseline, and a side on which to seek one of the other points to form a triangle, what is the locus of points of equal suitability such that the same connections result when approached from any direction?

For instance, consider the points A, B, C, and D:

```
    A
  /   
 /     
B------D
     /
     /  
    C
```

If AB is the baseline and D is more suitable than C, we want the point A to be more suitable than C when the baseline is BD. If this is the case, we can avoid checking to see if the most suitable triangle with baseline BD crosses element ABD. Since without this criterion every element must be checked for overlap, it is a substantial timesaver indeed.
The locus of points equally suitable to form a triangular element with \(AB\), must be symmetric about the perpendicular bisector of \(AB\) since the specification of A and B is orderless. In like fashion, the locus of points of equal suitability for \(BD\) is symmetric about its own perpendicular bisector. If the locus of points equally suitable for \(BD\) is different from those equally suitable for \(AB\), then it is possible to pick a point between the two loci such that one of the two baselines will prefer it to A or D. Therefore, they are the same locus. Being the same locus and symmetric about two arbitrary lines, the locus is a circle.

Therefore, the most suitable triangulation is considered to be the one which, when circumscribed by a circle, the centre of the circle is farthest behind the baseline. That is to say, the signed distance \(CD\) is a minimum (see Figure 4).

If any \(n\) nodes are found to lie on the circle defining the most suitable triangulation, they are all triangulated in right-hand order since any triangulation combinations would be equally satisfactory. A flowchart of the algorithm is given on the next page.
FLOWCHART OF TRIANGULAR MESH GENERATION ALGORITHM

* PICK BASELINE I-J *
* ON THE BOUNDARY AND *
* POINT N BEHIND IT. *

* **************

* FIND NS(M), M=1, IR *
* EQUALLY MOST SUIT-
* ABLE FOR I-J ON SIDE*
* OPPOSITE N AND NOT *
* CROSSING ANY BOUND-
* ary *

* **************

* REORDER NS(M) TO *
* PIVOT ON J *

* **************

* FOR M=1, IR:
* ADD ELEM I-J-NS(M) *
* STACK SIDE I-NS(M) *
* WITH BEHIND NODE J *
* N=1, I=NS(M) *

* **************

* IS *
* I-J A Y *
* BOUNDARY SEG-
* MENT? *

* **

* N *
* V *
* IS THE Y STACK *

* **

* IS *
* I-J IN Y *
* TWO ELEMENTS? **** *

* DONE }

**

* GET I,J,N OFF STACK *
* TOP AND DECREMENT *

**

**********

Figure 5
Example 3

NUMBER OF ELEMENTS = 158
NUMBER OF CORNER PTS = 121
NUMBER OF NODES = 399
Example 4
4.3 Algorithm to Generate Side Noded Connectivity

By renumbering the finite element nodes associated with a particular finite element mesh, the original sparse matrix equation forming the model may be converted to a banded matrix equation, which allows smaller computer storage requirements. The FEHPOL System uses an algorithm attributed to Collins (Ref. 11) to perform node renumbering for bandwidth minimization. Rather than feeding the algorithm side noded connectivity on which it would operate very inefficiently, an algorithm was developed for producing side noded connectivity from corner point connectivity while preserving the optimization of the numbering scheme. The algorithm is especially useful when used in conjunction with the Collins routine since the tables he sets up can be used. A flow chart of the algorithm is given on the next page and an example of its use is in Example 4.

4.4 The Internal Node Generation Algorithm

When modelling problems in continuous media it is often unnecessary to precisely locate each internal node. For regularly shaped regions (such as simple polygons) internal nodes may be generated using curvilinear coordinate mapping. For the highly irregular regions associated with coastal engineering problems, a more general yet equally automatic method is needed to generate internal nodes which reflect the local boundary density. An algorithm has been developed which is used in the FEHPOL System for this purpose.

Consider a simply connected polygon whose \( n \) vertices are finite element nodes. The linear node density at the \( i \)th node (with coordinates \( x_i, y_i \)) may be expressed as follows:
ALGORITHM TO GENERATE SIDE NODED CONNECTIVITY

* BANDWIDTH=0, NUM=0 *

* FOR L=1 TO NUMBER OF CORNER POINTS *
  * FIND NODE L *
  * NODERED Y *
    * CONNECTED TO L? *
    N *
  V *
  KKK=NUM=NUM+1 GIVE *
  L NEW NUMBER=NUM *
  V *
* FOR J=1 TO NUMBER OF NODES CONNECTED TO L *
  M= NODE NUMBER OF *
  JTH NODE CONNECTED TO NODE L *
  V *
  M < L? *
  V *
* FOR K=1, NUMBER OF NODES/SIDE *
  NUM=NUM+1 ADD SIDE *
  NODE ON SIDE L-M *
  GIVE IT NEW NUMBER= *
  ER=NUM *
  BANDWIDTH=NUM-KKK *
  +1 IF GREATER THAN *
  OLD BANDWIDTH *
  V *

---

Figure 6
**Example 5**

**Mesh Square with Three NODEd Triangles**

1. **Write Connectivity**
   - **Connectivity Table:**
     
     | 1 | 3 | 1 | 2 | 2 | 4 | 1 | 3 | 3 | 7 | 4 | 3 |
     |---|---|---|---|---|---|---|---|---|---|---|---|
     | 4 | 8 | 4 | 7 | 5 | 5 | 7 | 3 | 6 | 9 | 7 | 5 |

2. **Nodal Coordinates**
   - **X-Coord**
   - **Y-Coord**
     
     | 1  | 1.0000 | 1.0000 | 2 | 2.0000 | 1.0000 |
     | 3  | 2.0000 | 2.0000 | 4 | 1.0000 | 2.0000 |
     | 5  | 3.0000 | 2.0000 | 6 | 3.0000 | 1.0000 |
     | 7  | 2.0000 | 3.0000 | 8 | 1.0000 | 3.0000 |

3. **Number of Elements:** 8
4. **Number of Corner PTS:** 9
5. **Number of NODEs:** 25
6. **Bandwidth:** 15

**Mesh Square with Six NODEd Triangles**

1. **Write Connectivity**
   - **Connectivity Table:**
     
     | 1 | 9 | 1 | 5 | 3 | 2 | 6 | 2 | 13 | 1 | 9 | 4 | 3 | 10 |
     |---|---|---|---|---|---|---|---|---|---|---|---|---|---|
     | 3 | 21 | 13 | 9 | 14 | 10 | 11 | 4 | 24 | 13 | 21 | 15 | 14 | 22 |
     | 5 | 16 | 21 | 9 | 17 | 11 | 12 | 6 | 25 | 21 | 16 | 23 | 17 | 18 |
     | 7 | 5 | 16 | 9 | 7 | 12 | 6 | 8 | 20 | 16 | 5 | 19 | 7 | 8 |

2. **Nodal Coordinates**
   - **X-Coord**
   - **Y-Coord**
     
     | 1 | 1.0000 | 1.0000 | 2 | 1.5000 | 1.0000 |
     | 3 | 1.5000 | 1.5000 | 4 | 1.0000 | 1.5000 |
     | 5 | 2.0000 | 1.0000 | 6 | 2.0000 | 1.5000 |
     | 7 | 2.5000 | 1.5000 | 8 | 2.5000 | 1.0000 |
     | 9 | 2.0000 | 2.0000 | 10 | 1.5000 | 2.0000 |
     | 11 | 2.0000 | 2.5000 | 12 | 2.5000 | 2.0000 |
     | 13 | 1.0000 | 2.0000 | 14 | 1.5000 | 2.5000 |
     | 15 | 1.0000 | 2.5000 | 16 | 3.0000 | 2.0000 |
     | 17 | 2.5000 | 2.5000 | 18 | 3.0000 | 2.5000 |
     | 19 | 3.0000 | 1.5000 | 20 | 3.0000 | 1.0000 |
     | 21 | 2.0000 | 3.0000 | 22 | 1.5000 | 3.0000 |
     | 23 | 2.5000 | 3.0000 | 24 | 1.0000 | 3.0000 |
     | 25 | 3.0000 | 3.0000 | | | | |

3. **Number of Elements:** 8
4. **Number of Corner PTS:** 9
5. **Number of NODEs:** 25
6. **Bandwidth:** 15
where the node numbers are cyclic (i.e. \( \text{x}_{n+1} = \text{x}_1 \)). A suitable way of making the density of the internal nodes generated to reflect the local boundary density is:

\[
\sum_{i=1}^{n} \left( \frac{1}{(x-x_i)^2 + (y-y_i)^2} \right)
\]

\[
\int_{S} \frac{1}{\sqrt{(x_{i+1}-x_i)^2 + (y_{i+1}-y_i)^2} + \sqrt{(x_1-x_{i-1})^2 + (y_1-y_{i-1})^2}}
\]

since this is essentially a normalized integral, weighted at the point \( x, y \) by the inverse square of the distance to the boundary.

The algorithm divides the area into square cells whose sides have length equal to the average boundary segment length. Then it calculates the value of the density function at the centre of each of the cells. The value times the area of the cell gives the approximate number of nodes to place in that cell. To vary the total number of nodes in the region but reflect the boundary density a scale factor may be set by the user which scales the number of nodes per cell before converting to fixed point.
Once the number has been determined the nodes are placed in the cell in the following manner. The first node is placed at the centre of the cell and subsequent nodes \((i=2,n)\) are placed on a circle of diameter two thirds the width of the cell centred at the first node at an angle from an arbitrarily picked base of \(2^i\). For instance, if there are to be three nodes in a cell, the first node is placed at the centre of the cell, the second node is placed at four radians past the base angle, and the third is placed at nine radians past the base angle. This ensures sparse distribution and, since the number of nodes in a practical problem for each cell ranges in the small whole numbers, relatively acceptable finite elements. Once a finite element mesh is generated, it is possible to smooth the mesh very easily by locating each generated node at the average position of itself and nodes connected to it.

This algorithm has proven to provide a reasonable internal node distribution with a minimum of computation and without user specified zoning or local density values. As such it is ideally suited to use in the FEHPOL System, although probably not flexible enough to be very attractive for stand alone use. In Figure 5 an example of the use of the algorithm is given showing a mesh whose internal nodes were generated using it.

4.5 The Contour Plotting Algorithm

To help display the solution values generated by a particular finite element running under the FEHPOL System, a contour plotting algorithm was developed which uses the shape function of the element, rather than finite difference interpolation, to locate and draw contour lines. The algorithm works for any element type and for any number of contour levels.
The Boundary

Generated Internal Nodes

Example 6
The routine is table driven, that is to say, it uses a sub-connectivity table, a table of sub-nodal coordinates (in homogenous form), and a table of shape functions for the various types of finite elements. By using these tables, drawing the contours for any element type can be reduced to drawing contours on several linear, triangular finite elements. The values of the state variables being plotted are calculated at each sub-node in every element. The coordinates of the sub-nodes are calculated from the coordinates of the elements nodes and the homogenous coordinate table. Once these values are available it is a triviality to generate straight line contours on each sub-element. Both a flowchart of the algorithm's logic, and a listing of the FEHPOL implementation are given in the next few pages.
Algorithm to Plot Contours on Finite Element Region
Using the Shape Function of the Elements

* Do for every finite element
* Do for every contour level
* Do for number of subnodes/element

- Find X,Y coordinates of the subnodes using the table of homogenous coordinates
- Find the value of the function at each subnode using the shape function for this type of element

* Do for number of sub-elements/element
- Call linear three node triangle contour plotting routine for the ith sub-element using subconnectivity table

Figure 8
4.6 Smooth Interpolation

When trying to represent irregularly shaped boundaries, it can be extremely useful to be able to interpolate points along a smooth curve joining an existing set of points. In graphical analysis, various sorts of function approximations are often used to define a smooth curve fitting a particular set of data points which are known. To perform curve manipulation in FEH POL a more general technique was used.

Consider a set of $n$ ordered points $(x_i, y_i)$ through which we wish to approximate a smooth curve. For linear interpolation we use the homogeneous coordinate $t$ (usually without realizing it), to define interpolated points as follows:

$$\begin{align*}
x(t) &= (t-i)x_i + (1+i-t)x_{i+1} \quad \text{if } t < i+1 \\
y(t) &= (t-i)y_i + (1+i-t)y_{i+1} \quad \text{if } t < i+1
\end{align*}$$

ie. $x(3.5) = (3.5-3.0)x_3 + (1+3-3.5)x_4 = (x_3 + x_4)/2$

By considering further points to determine the interpolation on each segment a higher order interpolation is reached:

$$x(t) = f(x_{i-1}, x_i, x_{i+1}, x_{i+2}, t) \quad \text{if } t < i+1$$

Since $x(i-1) = x_{i-1}$,

$$x(t) = x_{i-1} + (t-i-1) [g(x_{i-1}, x_i, x_{i+1}, x_{i+2}, t)]$$

and since $x(i) = x_i$,

$$x(t) = x_{i-1} + (t-i-1) [(x_i - x_{i-1}) + (t-i)h(x_{i-1}, x_i, x_{i+1}, x_{i+2}, t)]$$
Carrying this to its logical conclusion:

\[
x(t) = x_{i-1} + (t+i-1) * ((x_i - x_{i-1}) + (t-i) * ((x_{i-1} - 2x_i + x_{i+1})/2) +
(t-i-1) * ((-x_{i-1} + 3x_i - 3x_{i+1} + x_{i+2})/6))
\]

and

\[
y(t) = y_{i-1} + (t+i-1) * ((y_i - y_{i-1}) + (t-i) * ((y_{i-1} - 2y_i + y_{i+1})/2) +
(t-i-1) * ((-y_{i-1} + 3y_i - 3y_{i+1} + y_{i+2})/6))
\]

The end segments present a slight difficulty, since neither \(x_0\) nor \(x_{n+1}\) is defined. This is remedied (in the FEHPOL System) by defining \(x_0 = x_1\) and \(x_{n+1} = x_n\) which simply straightens the curve slightly at the ends. Since no shape information at the end may be deduced from the original curve this is the logical choice.

The interpolation types, SMOOTH and REGULAR, differ only in the relative density of distribution of the points. REGULAR interpolation causes the new nodes to be spread as evenly as possible among the old nodes, while SMOOTH interpolation creates an entirely new set of points whose density along the curve reflect the density of points along the original curve. Examples 8 and 9 show the two interpolation types operating on a curve. In Appendix B several more examples of SMOOTH and REGULAR interpolation are given. This method produces extremely well conditioned interpolations for sharp corners, multiply specified points, and curves which double back on themselves, as well.
Example 8
SUBROUTINE FPMESH(S1,S2,S3,AA,CY,NS,NR,X,Y,M1,M2,NC,1NL,NB,NP)

SEGMENT NAME........................................FMESH
SEGMENT LENGTH IN CARDS.........................126
DATE OF LAST ALTERATION.......................21/07/76
NUMBER OF LOCAL INTEGERS........................18
NUMBER OF LOCAL REALS..........................13

FEHPOL ROUTINES CALLED..........................FPSSOL

FORTRAN LIBRARY ROUTINES CALLED................SQRT
..IABS
OTHER LIBRARY ROUTINES CALLED...................NONE

CALLING ROUTINES.................................FPCNCT

COMMONS REFERENCED...............................NONE

REMARKS:
THIS ROUTINE PERFORMS THE CONNECTION OF A FINITE
ELEMENT MESH GIVEN THE LOCATION OF ALL THE NODES
AND IDENTIFICATION OF THE BOUNDARY NODES.

ARGUMENTS:
S1,S2,S3=WORKAREA FOR STACKS, LENGTH AT LEAST M1
AA=WORKAREA, LENGTH AT LEAST M2
NS=WORKAREA, LENGTH AT LEAST M2
NS,NR=WORKAREAS, LENGTH AT LEAST M2
X,Y=NODE COORDINATES, DIMENSION NC
M1=MAXIMUM NUMBER OF ELEMENTS ALLOWED
M2=MAX NUMBER OF EQUALLY ATTRACTIVE CONNECTIONS FOR
ANY BASELINE. FOR ANY REASONABLE MESH M2=20
NC,NL,NB=NUM OF CORNER POINTS, ELEMS, BOUNDARIES
NP(I)=NUMBER OF NODES ON THE ITH BOUNDARY

FLOWCHART OF TRIANGULAR MESH GENERATION
ALGORITHM

******************#******
* PICK BASELINE I-J *
* ON THE BOUNDARY AND *
* POINT N BEHIND IT. *
**************

FIND NS(M),M=1,IR

EQUALITY MOST SUIT-
ABLE FOR I-J ON SIDE
OPPOSITE N AND NOT
CROSSING ANY BOUND-
ARY

**************
IF(IL.GT.J)GO TO 1020
IF(IL.GT.I)GO TO 1020
IF(J.GT.IT)GO TO 1020
IF(I.GT.IT)GO TO 1020
IF(IABS(I-J).EQ.1)GO TO 120
IF(IABS(I-J).EQ.NP(MM)-1)GO TO 120
1020 IL=IT+1
NU=0
DO 40 L=1,NG
II=CY(1,L)
JJ=CY(2,L)
KK=CY(3,L)
IF(I.NE.II.AND.I.NE.JJ.AND.I.NE.KK)GO TO 40
IF(J.NE.II.AND.J.NE.JJ.AND.J.NE.KK)GO TO 40
NU=NU+1
IF(NU.GE.2)GO TO 120
40 CONTINUE
35 XI=X(I)
XJ=X(J)
XN=X(N)
YI=Y(I)
YJ=Y(J)
YN=Y(N)
DSQ=((XJ-XI)**2+(YJ-YI)**2)/4.0
DO 60 L=1,NC
IF(L.EQ.I)GO TO 60
IF(L.EQ.J)GO TO 60
XL=X(L)
YL=Y(L)
IF(FPSSOL(XI,YI,XJ,YJ,XL,YL,XN,YN))80,80,60
80 CALL FP CNTR(XI,YI,XJ,YJ,XL,YL,XC,YC,RSQ)
R=RSQ-DSQ
IF(FPSSOL(XI,YI,XJ,YJ,XC,YC,XL,YL))314,314,313
314 R=-R
313 IF(R.GT.RN)GO TO 60
C * CHECK TO SEE IF SEGMENT I-L CROSSES BOUNDARY. *
C * IF YES, GO TO 60 *
C IT = 0
IL=1
DO 1030 MM=1,NB
IT=IT+NP(MM)
DO 95 NO=IL,IT
N1=NO+1
IF(NO.EQ.IT)N1=IL
IF(NO.EQ.I)GO TO 95
IF(NO.EQ.L)GO TO 95
IF(N1.EQ.I)GO TO 95
IF(N1.EQ.L)GO TO 95
IF(FPSSOL(XI,YI,XL,YL,X(NO),Y(NO),X(N1),Y(N1)))95,96,95
96 IF(FPSSOL(X(NO),Y(NO),X(N1),Y(N1),XI,YI,XL,YL))95,60,95
95 CONTINUE
C * CHECK TO SEE IF SEGMENT J-L CROSSES BOUNDARY. *
C * IF YES, GO TO 60 *
C
DO 97 NO=IL, IT
N1=NO+1
IF(NO.EQ.IT)N1=IL
IF(NO.EQ.J)GO TO 97
IF(NO.EQ.L)GO TO 97
IF(N1.EQ.J)GO TO 97
IF(N1.EQ.L)GO TO 97
IF(FPSSOL(XJ,YJ,XL,YL,X(NO),Y(NO),X(N1),Y(N1)))97,98,97
98 IF(FPSSOL(X(NO),Y(NO),X(N1),Y(N1),XJ,YJ,XL,YL))97,60,97
97 CONTINUE
1030 IL=IT+1
IF(R.LT.RN)GO TO 90
IR=IR+1
NS(IR)=L
GO TO 60
90 CONTINUE
RN=R
IR=1
NS(IR)=L
60 CONTINUE
IF(IR.GT.M2)GO TO 920
DO 30 II=1,IR
IO=NS(II)
30 AA(II)=D(XJ,YJ,X(IO),Y(IO))-D(XI,YI,X(IO),Y(IO))
DO 25 II=1,IR
IO=0
DO 20 JJ=1,IR
IF(AA(JJ).GE.AA(II))IO=IO+1
20 CONTINUE
25 NR(II)=NS(II)
DO ,10 II=1,IR
NG=NG+1
SP=SP+1
S1(SP)=I
S2(SP)=NR(II)
S3(SP)=J
C ******************************************************************************
C * ADD NEW ELEMENT I-J-NR(II) TO CONNECTIVITY TABLE *
C ******************************************************************************
CY(1,NG)=I
CY(2,NG)=J
CY(3,NG)=NR(II)
N=I
10 I=NR(II)
GO TO 1000
C ******************************************************************************
C * UNSTACK BASELINE AND BEHIND NODE (I,J) AND (N) *
C ******************************************************************************
120 I=S1(SP)
J=S2(SP)
N=S3(SP)
SP=SP-1
IF(SP)920,1000,1000
920 NL=NG
RETURN
END
THE SHAPE FUNCTION FOR THIS
TREAT EACH SUBNODE USING
FIND THE VALUE OF THE FUNC-

A

HOMOGENOUS COORDINATES
SUBNODES USING THE TABLE
FIND X,Y COORDINATES OF THE

-------------------------------
DO FOR NUMBER OF SUBNODES/ELEMENT

-------------------------------
DO FOR EVERY CONTOUR LEVEL

-------------------------------
DO FOR EVERY FINITE ELEMENT

-------------------------------

ALGORITHM TO PLOT CONTOURS ON FINITE ELEMENT REGION

ELEMENT IN AN ELEMENT JOINS OTHERS.
SUBNODES (THAT IS THE POINTS AT WHICH EACH SUB-
NODES BEING PLotted AT EACH OF THE
SHAPE FUNCTION OF THE ELEMENT TO FIND THE VALUES
PLOTTING WHICH PCOMP CAN PERFORM. IT USES THE
ORDER ELEMENTS TO REDUCE THE CONTOUR PLOTTING
THIS ROUTINE USES THE SUBCOORDINATIVITY OF HIGHER

REMARKS:

MEXECA
COMMONS REFERENCED.
FRENCH
CALLING ROUTINES
OTHER LIBRARY ROUTINES CALLED:
PORTHNAV LIBRARY ROUTINES CALLED:
FEHPOL ROUTINES CALLED:
6
3
16
2/1/76
104
SUBROUTINE RCPL
SUBROUTINE RCPL(BOUNDARY,NUMBER_INDEX)
REAL CONLEV(NUMLEV),F(1),XX(15),YY(15),FF(15)
INTEGER CON6NT(3,16)
REAL H6NT(3,15)
INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
COMMON/MSAR6A/CY,NT,VM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NE(9)
EQUIVALENCE (Y(300),F(1))
DATA CON6NT(1,1),CON6NT(2,1),CON6NT(3,1) / 1, 7, 12/
1 CON6NT(1,2),CON6NT(2,2),CON6NT(3,2) / 7, 12, 13/
2 CON6NT(1,3),CON6NT(2,3),CON6NT(3,3) / 13, 12, 6/
3 CON6NT(1,4),CON6NT(2,4),CON6NT(3,4) / 6, 13, 15/
4 CON6NT(1,5),CON6NT(2,5),CON6NT(3,5) / 15, 11, 6/
5 CON6NT(1,6),CON6NT(2,6),CON6NT(3,6) / 11, 15, 10/
6 CON6NT(1,7),CON6NT(2,7),CON6NT(3,7) / 10, 3, 11/
7 CON6NT(1,8),CON6NT(2,8),CON6NT(3,8) / 5, 10, 15/
8 CON6NT(1,9),CON6NT(2,9),CON6NT(3,9) / 5, 15, 14/
9 CON6NT(1,10),CON6NT(2,10),CON6NT(3,10) / 14, 15, 13/
1 CON6NT(1,11),CON6NT(2,11),CON6NT(3,11) / 13, 14, 4/
2 CON6NT(1,12),CON6NT(2,12),CON6NT(3,12) / 4, 13, 7/
3 CON6NT(1,13),CON6NT(2,13),CON6NT(3,13) / 4, 8, 14/
4 CON6NT(1,14),CON6NT(2,14),CON6NT(3,14) / 14, 8, 9/
5 CON6NT(1,15),CON6NT(2,15),CON6NT(3,15) / 9, 14, 5/
6 CON6NT(1,16),CON6NT(2,16),CON6NT(3,16) / 2, 8, 9/
DATA H6NT(1,1),H6NT(2,1),H6NT(3,1) / 1.0,0.0,0.0/
1, H6NT(1,2),H6NT(2,2),H6NT(3,2) / 0.0,1.0,0.0/
H6NT(1,3), H6NT(2,3), H6NT(3,3) / 0.0, 0.0, 1.0/
H6NT(1,4), H6NT(2,4), H6NT(3,4) / 0.0, 0.5, 0.0/
H6NT(1,5), H6NT(2,5), H6NT(3,5) / 0.0, 0.0, 0.5/
H6NT(1,6), H6NT(2,6), H6NT(3,6) / 0.5, 0.0, 0.5/
H6NT(1,7), H6NT(2,7), H6NT(3,7) / 0.75, 0.25, 0.0/
H6NT(1,8), H6NT(2,8), H6NT(3,8) / 0.25, 0.75, 0.0/
H6NT(1,9), H6NT(2,9), H6NT(3,9) / 0.0, 0.75, 0.25/
H6NT(1,10), H6NT(2,10), H6NT(3,10) / 0.0, 0.25, 0.75/
H6NT(1,11), H6NT(2,11), H6NT(3,11) / 0.25, 0.0, 0.75/
H6NT(1,12), H6NT(2,12), H6NT(3,12) / 0.75, 0.0, 0.25/
H6NT(1,13), H6NT(2,13), H6NT(3,13) / 0.5, 0.25, 0.25/
H6NT(1,14), H6NT(2,14), H6NT(3,14) / 0.25, 0.5, 0.25/
H6NT(1,15), H6NT(2,15), H6NT(3,15) / 0.25, 0.25, 0.5/

\[
\text{PHI6NT(EL1,EL2,EL3)} = F(II)\times\text{EL1}(2.0\times\text{EL1}-1.0) + F(JJ)\times\text{EL2}(2.0\times\text{EL2}-1.0) + F(KK)\times\text{EL3}(2.0\times\text{EL3}-1.0) + F(IJ)\times4.0\times\text{EL1}\times\text{EL2} + F(JK)\times4.0\times\text{EL2}\times\text{EL3} + F(KI)\times4.0\times\text{EL3}\times\text{EL1}
\]

DO 10 I=1,NN
10 F(I)=PL(PT-INDX-I)
DO 20 I=1,NL
II=CY(1,I)
JJ=CY(2,I)
KK=CY(3,I)
LL=CY(4,I)
IJ=LL
JK=CY(5,I)
KI=CY(6,I)
DO 30 J=1,NUMLEV
C=CDNLEV(J)
GO TO (100,100,300,100,100,600,100),IE
100 CALL FPSTOP(53)
300 CALL FP CNP3(X,Y,F,II,JJ,KK,C)
GO TO 30
600 DO 40 K=1,15
XX(K)=X(II)\times\text{H6NT}(1,K) + X(JJ)\times\text{H6NT}(2,K) + X(KK)\times\text{H6NT}(3,K)
YY(K)=Y(II)\times\text{H6NT}(1,K) + Y(JJ)\times\text{H6NT}(2,K) + Y(KK)\times\text{H6NT}(3,K)
40 FF(K)=P6NT(H6NT(1,K),H6NT(2,K),H6NT(3,K))
DO 610 K=1,16
III=CON6NT(1,K)
JJJ=CON6NT(2,K)
KKK=CON6NT(3,K)
610 CALL FP CNP3(XX,YY,FF,III,JJJ,KKK,C)
GO TO 30
30 CONTINUE
20 CONTINUE
RETURN
END
CHAPTER 5

EXAMPLES
5.1 **FEHPOL Solution Programs**

The FEHPOL system supervises finite element programs. The solution programs are essentially 'black boxes' which represent given partial differential equations. The system inputs finite element mesh descriptions, boundary conditions, and constants to the solution program and it returns the values of various state variables at finite element nodes. While several solution programs may be combined or run under the FEHPOL system in parallel, the orientation towards minicomputer environments makes the supervision of a single solution program the most likely configuration. In practice, solution programs are overlayed over part of the FEHPOL system.

At present, two existing finite element solution programs have been modified to run under the system. Their solution capabilities include:

1. **DISPERSION EQUATION SOLUTION PROGRAM**
   a) Transport or Heat Transfer Equation for One Variable (Time Independent)
   b) Perfect Flow, Potential Formulation (T.I.)
   c) Confined Seepage Equation (T.I.)
   d) Potential Formulation of Shallow Water Problems (T.I.)

2. **NAVIER-STOKES EQUATION SOLUTION PROGRAM**
   a) Navier Stokes Equation for Incompressible Viscous Fluids, Including Time Dependence

In retrieving elemental properties, boundary conditions and any other parameters from the FEHPOL data structure, each solution program defines the corresponding names. This means that the
solution parameters to be set (using the FEHPOL SET command) depend on the supervised program.

5.2 The Solution Program DSPSTD

The solution program DSPSTD uses the finite element method to solve the following differential equation:

\[ c \left( \frac{\partial^2 \psi}{\partial x^2} + K_2 \frac{\partial^2 \psi}{\partial y^2} - U \frac{\partial \psi}{\partial x} - V \frac{\partial \psi}{\partial y} - \beta \psi \right) + \sum \frac{P_i}{A_i} + \sum Q_i A_i = 0 \]

The various solution parameters associated with the equation and their default values are given in the following table.

<table>
<thead>
<tr>
<th>Item in equation</th>
<th>FEHPOL parameter name</th>
<th>Type</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \psi )</td>
<td>CONCENTRATION</td>
<td>nodal</td>
<td>none</td>
</tr>
<tr>
<td>C</td>
<td>HEAT-COEFF</td>
<td>global</td>
<td>1.00</td>
</tr>
<tr>
<td>K_1</td>
<td>DX</td>
<td>elemental</td>
<td>1.00</td>
</tr>
<tr>
<td>K_2</td>
<td>DY</td>
<td>elemental</td>
<td>1.00</td>
</tr>
<tr>
<td>( \beta )</td>
<td>DECAY</td>
<td>global</td>
<td>0.00</td>
</tr>
<tr>
<td>P_i</td>
<td>DISCHARGE</td>
<td>nodal</td>
<td>0.00</td>
</tr>
<tr>
<td>Q_i</td>
<td>GLOBAL-DISCHARGE</td>
<td>elemental</td>
<td>0.00</td>
</tr>
<tr>
<td>U</td>
<td>X-VELOCITY</td>
<td>nodal</td>
<td>0.00</td>
</tr>
<tr>
<td>V</td>
<td>Y-VELOCITY</td>
<td>nodal</td>
<td>0.00</td>
</tr>
</tbody>
</table>

The first example of the use of this solution program is a simple rectangular channel problem. The concentration boundary condition has been set at either end of the channel and a decay constant of 0.01 has been specified. A contour plot of the resulting levels of concentration, the finite element mesh generated and the commands is given on the next page.
CURVE 4,0 24,0:TOP
CURVE 24,2 4,2:BOT
INTERPOLATE TOP NODES=10,LINEAR
INTERPOLATE BOT NODES=10,LINEAR
CURVE TOP(10) BOT(1):END1
CURVE TOP(1) BOT(10):END2
REGION TOP BOT:CHANNEL
MESH CHANNEL WITH SIX NODED TRIANGLES
SET CONCENTRATION=100 ON END1, CONCENTRATION=0 ON END2, DECAY=0.01
EXECUTE "DISPERSION EQUATION SOLUTION"
SET XMAX=25, YMAX=2
PLOT CONTOUR MAP OF SOLUTION, LEVELS=(10, 20, 30, 40, 50, 60, 70, 80, 90)
PLOT AND LABEL MESH

Example 10
The following example illustrates the use of FEHPOL to form a model of the dispersion of sewage which is released into the river Guaíba from the city of Porto Alegre in Rio Grande do Sul, Brazil. For the example which follows, no velocities were introduced, however since the sewage concentration is non-conservative, the decay coefficient was set. The concentration boundary condition is set on the mouth of the river, and a discharge at a node representing the outfall point. Notice that in the command sequence it is not necessary for the user to know anything about the finite element mesh which was generated except to specify an element type in the MESH statement.

Plotting is only one of the ways of displaying the results of the finite element calculations. As this example shows, any FEHPOL parameter (including those passed back from the solution program via the FPSPUT supervisor call) may be printed at the user's discretion, as well as other sorts of intermediate information such as the mesh specifics, the connectivity table, nodal coordinates, and representations of REGIONS, CURVES, and NODEs. In this example the mesh specifics and the solution (concentration) vector have been requested and printed.
NODE 13,47: PTA-DA-CADEIA
NODE 15,15: SALGADO
CURVE 25,0 40,3: LAGOA-DOS-PATOS
INTERPOLATE LAGOA-DOS-PATOS NODES=3, LINEAR
CURVE LAGOA-DOS-PATOS(3)
   35,6 34,9 37,12 38,16 35,18 32,17 32,22 28,19
   25,20 23,17 21,18 22,21 18,21 19,24 17,25 16,28
   10,28 12,30 12,33 10,35 9,38 10,40 13,43
PTA-DA-CADEIA: ALEGRE
CURVE 11,48 10,45 7,45 3,42 1,40 2,36 1,30
   1,23 4,22 5,19 5,14 10,13 SALGADO 15,5,9 18,4
   22,6 27,8 28,6 26,3: SACOS
REGION SACOS LAGOA-DOS-PATOS ALEGRE: RIO-GUAIBA
LET 7,41 5,35 7,5,33 11,21 21,12 = INTERNAL NODES FOR RIO-GUAIBA
MESH RIO-GUAIBA WITH SIX NODE TRIANGLES
SET XMAX=50, YMAX=60
PLOT MESH
SET DISCHARGE=1000 ON PTA-DA-CADEIA,
   CONCENTRATION=150 ON LAGOA-DOS-PATOS,
   DECAY = 0.001
EXECUTE "DISPERSION EQUATION SOLUTION WITH DECAY=0.001"
WRITE MESH SPECS, SOLUTION
PLOT CONTOUR MAP OF SOLUTION, LEVELS=(133, 167, 200, 233, 267, 300,
   333, 367, 400)
SET DECAY = 0.01
EXECUTE "DISPERSION EQUATION SOLUTION WITH DECAY=0.01"
PLOT CONTOUR MAP OF SOLUTION, LEVELS=(133, 167, 200, 233, 267, 300,
   333, 367, 400)
/*

Example 11
SOLUTION

<table>
<thead>
<tr>
<th>NUMBER OF ELEMENTS</th>
<th>54</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF CORNER PTS</td>
<td>51</td>
</tr>
<tr>
<td>NUMBER OF NODES</td>
<td>155</td>
</tr>
<tr>
<td>BANDWIDTH</td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>N/E</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF NODES</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td>NUMBER OF CORNER PTS</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>NUMBER OF ELEMENTS</td>
<td>139</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SOLUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0.448401E 04</td>
</tr>
<tr>
<td>4 0.542896E 03</td>
</tr>
<tr>
<td>7 0.365561E 04</td>
</tr>
<tr>
<td>10 0.311321E 04</td>
</tr>
<tr>
<td>13 0.332795E 04</td>
</tr>
<tr>
<td>16 0.268079E 04</td>
</tr>
<tr>
<td>19 0.264961E 04</td>
</tr>
<tr>
<td>22 0.254387E 04</td>
</tr>
<tr>
<td>25 0.245415E 04</td>
</tr>
<tr>
<td>28 0.255939E 04</td>
</tr>
<tr>
<td>31 0.216843E 04</td>
</tr>
<tr>
<td>34 0.190399E 04</td>
</tr>
<tr>
<td>37 0.199633E 04</td>
</tr>
<tr>
<td>40 0.196103E 04</td>
</tr>
<tr>
<td>43 0.171342E 04</td>
</tr>
<tr>
<td>46 0.152392E 04</td>
</tr>
<tr>
<td>49 0.171002E 04</td>
</tr>
<tr>
<td>52 0.181837E 04</td>
</tr>
<tr>
<td>55 0.140921E 04</td>
</tr>
<tr>
<td>58 0.127161E 04</td>
</tr>
<tr>
<td>61 0.142306E 04</td>
</tr>
<tr>
<td>64 0.145451E 04</td>
</tr>
<tr>
<td>67 0.120005E 04</td>
</tr>
<tr>
<td>70 0.106303E 04</td>
</tr>
<tr>
<td>73 0.105821E 04</td>
</tr>
<tr>
<td>76 0.118070E 04</td>
</tr>
<tr>
<td>79 0.114475E 04</td>
</tr>
<tr>
<td>82 0.101319E 04</td>
</tr>
<tr>
<td>85 0.825178E 03</td>
</tr>
<tr>
<td>88 0.968948E 03</td>
</tr>
<tr>
<td>91 0.104166E 04</td>
</tr>
<tr>
<td>94 0.774751E 03</td>
</tr>
<tr>
<td>97 0.838361E 03</td>
</tr>
<tr>
<td>100 0.594866E 03</td>
</tr>
<tr>
<td>103 0.661030E 04</td>
</tr>
<tr>
<td>106 0.853328E 03</td>
</tr>
<tr>
<td>109 0.543247E 03</td>
</tr>
<tr>
<td>112 0.610768E 03</td>
</tr>
<tr>
<td>115 0.487629E 03</td>
</tr>
<tr>
<td>118 0.358623E 03</td>
</tr>
<tr>
<td>121 0.472068E 03</td>
</tr>
<tr>
<td>124 0.538733E 03</td>
</tr>
<tr>
<td>127 0.413379E 03</td>
</tr>
<tr>
<td>130 0.339824E 03</td>
</tr>
<tr>
<td>133 0.379575E 03</td>
</tr>
<tr>
<td>136 0.214023E 03</td>
</tr>
<tr>
<td>139 0.399686E 03</td>
</tr>
<tr>
<td>142 0.418970E 03</td>
</tr>
<tr>
<td>145 0.193008E 03</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>148</td>
</tr>
<tr>
<td>152</td>
</tr>
</tbody>
</table>

**NUMBER OF PARAMETERS**: 8

**PARAMETER SPACE LEFT**: 967

Example 11 (cont'd)
Example 11 (cont'd)
Example 11 (cont'd)
The following example models seepage around a dam in a confined region. Extended subscript notation is used to define the curve HEAD, which consist of the third through the fifth points representing the curve AAA. In this example, the DISPLAY stream was assigned to the same physical device as the source input (at FEHPOL system initialization time), so that when the FEHPOL command "WRITE SOLUTION" was issued, the response came on the same device. For most applications involving the use of any hardcopy terminal device, such as a teletype, this is the logical assignment. Note that when writing the parameter pool or any entry in it, the current size and amount of free space left is also given in the response. The symbol pool size and current space available is given when a CURVE, REGION, or NODE, or the whole symbol pool write is issued.
CURVE 9.10 0,10 AAA
CURVE 20.10 11,10 BBB
INTERPOLATE AAA NODES=5,LINEAR
INTERPOLATE BBB NODES=5,LINEAR
CURVE AAA(5) 0,0 20,0 BBB(1) CCC
INTERPOLATE CCC NODES=9,LINEAR
REGION 9,5 AAA CCC BBB 10,5,AA=REGION1
LET 5,4 15,4 INTERKAL NODES FOR REGION1
MESH REGION1 WITH SIX NODED TRIANGLES
CURVE AAA(3-5) HEAD
SET CONCENTRATION=1000 ON HEAD
SET CONCENTRATION=0 ON BBB
SET OPTION=1 DX=1 ON REGION1, DY=1 ON REGION1, DEPTH=1 ON REGION1
EXECUTE SEEPAGE TEST PROBLEM
WRITE SOLUTION

PARAMETER TABLE: N/E VALUE N/E VALUE N/E VALUE
1 0.0.100000E 04 2 0.832816E 03 3 0.989222E 03
4 0.100000E 04 5 0.733073E 03 6 0.694651E 03
7 0.657040E 03 8 0.802855E 03 9 0.854577E 03
10 0.792335E 03 11 0.746734E 03 12 0.856199E 03
13 0.751545E 03 14 0.810000E 03 15 0.180806E 04
16 0.896971E 03 17 0.100000E 04 18 0.681549E 03
19 0.590148E 03 20 0.613349E 03 21 0.732195E 03
22 0.544442E 03 23 0.459259E 03 24 0.478214E 03
25 0.735842E 03 26 0.801765E 03 27 0.892979E 03
28 0.769846E 03 29 0.752905E 03 30 0.803899E 03
31 0.100000E 04 32 0.461556E 03 33 0.443694E 03
34 0.334783E 03 35 0.373895E 03 36 0.310806E 03
37 0.269371E 03 38 0.441166E 03 39 0.151908E 03
40 0.245854E 03 41 0.235520E 03 42 0.208356E 03
43 0.196603E 03 44 0.190524E 03 45 0.150590E 03
46 0.955622E 02 47 0.104855E 02 48 0.115691E 03
49 0.000000E 00 50 0.000000E 00 51 0.000000E 00
52 0.000000E 00 53 0.185307E 03 54 0.171055E 03
55 0.129417E 03 56 0.713308E 02 57 0.701017E 02
58 0.000000E 00 59 0.000000E 00 60 0.000000E 00
61 0.000000E 00 62 0.000000E 00

NUMBER OF PARAMETERS= 6
PARAMETER SPACE LEFT= 788

PLOT CONTOUR MAP OF SOLUTION, LEVELS=(100,200,300,400,500,
600,700,800,900)

Example 12
The following example shows the setup and solution of several problems on a multiply connected region. It demonstrates the facility of the mesh generation algorithm for such regions, and also the ease of modification of an existing problem description interactively and with graphical feedback. The region shown may represent the area immediately surrounding a three pillared offshore oil drilling platform or alternatively around a set of exhaust stacks from a factory depending on the boundary conditions applied and the interpretation given them.

In the first case, no boundary conditions were specified on the three internal boundaries. The concentration condition was set are on the opposing sides (named A and C). The value of the parameter CONCENTRATION is 1000 and 100 respectively. An interpretation of this problem is seepage past a set of circular non-porous tubes (cables, pipes). The resultant solution is plotted in a contour map on page 71.

The second set of boundary conditions adds the requirement that the value on the state variable be constant on each of the internal boundaries. They were set at the values that would be the solution of the problem at the centres in their absence. An interpretation of this problem could be with the state variable, CONCENTRATION, representing velocity potential. The contour map plotted on page 72 would then represent normals to the flow.

In the third step, the outside boundary was left free with the internal boundaries fixed. In the final example the internal boundaries had the concentration boundary condition set equal to 1000.0 and the outside boundary set to 0.0. The solution could represent the dispersion of an air pollutant from a set of three factory stacks, particularly since the decay term was set.
CURVE 100,200 100,800:A
CURVE A(2) 900,800:B
CURVE B(2) 900,200:C
CURVE C(2) A(1):D
CURVE 350,375 450,400 350,425:PILLAR1
CURVE 450,575 550,600 450,625:PILLAR2
CURVE 575,400 675,425 575,450:PILLAR3
INTER A NODES=5,LINEAR
INTER B NODES=7,LINEAR
INTER C NODES=5,LINEAR
INTER D NODES=7,LINEAR
INTER PILLAR1 NODES=7,CIRCULAR
INTER PILLAR2 NODES=7,CIRCULAR
INTER PILLAR3 NODES=7,CIRCULAR
REGION A B C D:ABCD
MESH ABCD HOLES=(PILLAR1,PILLAR2,PILLAR3) WITH SIX NODED TRIA
PLOT AND LABEL MESH
SET XMAX=1200,XMAX=1000
KEEP 1
SET CONCENTRATION=1000 ON A,CONCENTRATION=100 ON C
EXECUTE "EXAMPLE 13A"
PLOT CONTOUR MAP OF SOLUTION,LEVELS=(150,200,250,300,350,400,450
500,550,600,650,700,750,800,850,900,950)
SET CONCENTRATION=600 ON PILLAR1,CONCENTRATION=500 ON PILLAR2,
CONCENTRATION=400 ON PILLAR3
EXECUTE "EXAMPLE 13B"
PLOT CONTOUR MAP OF SOLUTION,LEVELS=(150,250,350,450,550,
650,750,850,950)
UNKEEP 1
SET CONCENTRATION=600 ON PILLAR1,CONCENTRATION=500 ON PILLAR2,
CONCENTRATION=400 ON PILLAR3
EXECUTE "EXAMPLE 13C"
PLOT CONTOUR MAP OF SOLUTION,LEVELS=(410,430,450,470,490,510,
530,550,570,590)
SET CONCENTRATION=1000 ON PILLAR1,CONCENTRATION=1000 ON PILLAR2,
CONCENTRATION=1000 ON PILLAR3,CONCENTRATION=1000 ON #97
SET CONCENTRATION=1000 ON #56,CONCENTRATION=1000 ON #102
SET CONCENTRATION=0 ON A,CONCENTRATION=0 ON B
SET CONCENTRATION=0 ON C,CONCENTRATION=0 ON D
SET DECAY=0.001
EXECUTE "EXAMPLE 13D"
PLOT CONTOUR MAP OF SOLUTION,LEVELS=(100,200,300,400,500,500,
700,800,900)
Example 13 (cont'd)
Example 13b
Example 13c
5.2 The Solution Program NAVSTK

The solution program NAVSTK uses the finite element method to solve the Navier-Stokes equation (pressure and velocity formulation). This program is of a more specialized nature than the program discussed in Section 5.2. For a detailed description of its capabilities, see Appendix C. Among the specifiable parameters are the following:

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CYCLES</td>
<td>Global</td>
<td>0</td>
</tr>
<tr>
<td>PRINT</td>
<td>Global</td>
<td>0</td>
</tr>
<tr>
<td>ORDER</td>
<td>Global</td>
<td>1</td>
</tr>
<tr>
<td>WEIGHTED</td>
<td>Global</td>
<td>0</td>
</tr>
<tr>
<td>NU</td>
<td>Global</td>
<td>1</td>
</tr>
<tr>
<td>DT</td>
<td>Global</td>
<td>10.0</td>
</tr>
<tr>
<td>CLENGTH</td>
<td>Global</td>
<td>1.0</td>
</tr>
<tr>
<td>UBAR</td>
<td>Global</td>
<td>1.0</td>
</tr>
<tr>
<td>ODIAM</td>
<td>Global</td>
<td>1.0</td>
</tr>
<tr>
<td>K3</td>
<td>Elemental</td>
<td>0.0</td>
</tr>
<tr>
<td>ENT</td>
<td>Nodal</td>
<td>none</td>
</tr>
<tr>
<td>US</td>
<td>Nodal</td>
<td>none</td>
</tr>
<tr>
<td>UN</td>
<td>Nodal</td>
<td>none</td>
</tr>
<tr>
<td>STATIC</td>
<td>Global</td>
<td>0</td>
</tr>
<tr>
<td>P</td>
<td>Nodal</td>
<td>none</td>
</tr>
<tr>
<td>NOB</td>
<td>Global</td>
<td>0</td>
</tr>
</tbody>
</table>

In the following example, the original output channel which was used by the solution is used for output. In this instance, FEHPOL is acting simply as a preprocessor which generates finite element data for the program.
NODE 1,1:A
NODE 1,5:B
NODE 7,5:C
NODE 7,1:D
CURVE A B:AB
CURVE B C:BC
CURVE C D:CD
CURVE D A:DA
INTERPOLATE AB NODES=3,LINEAR
INTERPOLATE BC NODES=3,LINEAR
INTERPOLATE CD NODES=3,LINEAR
INTERPOLATE DA NODES=3,LINEAR
REGION AB BC CD DA:ABCD
NODES 2,3 6,3:F INTERPOLATE F NODES=5,LINEAR
LET F = INTERNAL NODES FOR ABCD
NODE AB(2):A2 NODE CD(2):C2
MESH ABCD WITH THREE NODED TRIANGLES
SET US=0 ON BC,UN=0 ON BC,ENT=2 ON BC
SET US=0 ON DA,UN=0 ON DA,ENT=2 ON DA
SET US=0 ON A2,UN=1 ON A2,ENT=2 ON A2.
SET P=0.1 ON C2,N0B=0
SET ODIAM=0,NU=1,CYCLES=30,ORDER=1,PRINT=1,STATIC=1,OPT=0
SET DT=0.001,UBAR=5,LENGTH=4
EXECUTE "NAVIER-STOKES EQUATION SOLUTION PROGRAM"**^****************************#******#****#****#**#**********

FINITE ELEMENT PROGRAM TO SOLVE NAVIER-STOKES EQUATIONS

**FINITE ELEMENT PROGRAM TO SOLVE NAVIER-STOKES EQUATIONS**

**CONTROLLING PARAMETERS**

NUMBER OF NODES ......................... 13
NUMBER OF CORNER NODES ................. 13
NUMBER OF ELEMENTS ...................... 16
SEMI-BANDWIDTH .......................... 7
NUMBER OF RESTRICTED NODES ............ 7
NUMBER OF TIME CYCLES ................... 30
PRINTOUT OF RESULTS.EVERY CYCLE=1 ETC.. 1

INTG. TECHNIQUE.1=HALF-STEP,2-4=RUNGK. 1
WHETHER SOL. IS WEIGHTED.0=NOT,1=WTD... 1
NUMBER OF PRESSURE BOUNDARY VALUES..... 1
DYNAMIC VISCOSITY(CM2/SEC)............... 1.00000
LENGTH OF TIME STEP(SECS).............. 0.00100
PRINTOUT OF CHAR. PARAMETERS............ 1
PRINTOUT OF PRESSURE AND VELOCITIES.... 1
WHETHER MESH IS OPTIMIZED (0=NO,1=YES). 0

**CHARACTERISTIC PARAMETERS**

***************

75a
CHARACTERISTIC LENGTH ....................... 4.00000
FREE STREAM VELOCITY ........................ 5.00000
OBJECT DIAMETER .............................. 0.00000
REYNOLDS NUMBER .............................. 20.00000

CO-ORDINATES OF NODES

***************

1 7.000000000  3.000000000
2 6.000000000  3.000000000
3 7.000000000  5.000000000
4 7.000000000  1.000000000
5 5.000000000  3.000000000
6 4.000000000  5.000000000
7 4.000000000  1.000000000
8 4.000000000  3.000000000
9 3.000000000  3.000000000
10 1.000000000  5.000000000
11 1.000000000  1.000000000
12 2.000000000  3.000000000
13 1.000000000  3.000000000

THREE NODE CONNECTIVITY

***************

AND VELOCITY DIFF. BOUNDARY CONDITIONS

***************

IBN-INTERNAL BOUNDARY NODE.

ELEM NODES IBN ELEM NODES IBN ELEM NODES IBN
1  9 11 7 0  2 12 11 9 0  3 13 11 12 0
4 10 13 12 0  5 9 10 12 0  6 6 10 9 0
7  8  6 9 0  8 5 6 8 0  9 2 6 5 0
10  3  6 2 0 11  1 3 2 0 12  4 1 2 0
13  5  4 2 0 14  7 4 5 0 15  8 7 5 0
16  9  7 8 0

DETAILS OF VELOCITY BOUNDARY CONDITIONS

***************

NODE NUMBER ANGLE TYPE UN US.
2   2

AX

3 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
4 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
6 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
7 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
10 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
11 0.000 1.000 BOTH VELOCITIES SPECIFIED 0.00000 0.000
13 1.000 0.000 BOTH VELOCITIES SPECIFIED 1.00000 0.000

INITIAL VALUES OF VELOCITY

***************
STATIC CONDITIONS
BOUNDARY DETAILS FOR PRESSURE

NUMBER OF PRESSURE BOUNDARY VALUES...

1

NODE NUMBER    PRESSURE VALUE

1       0.100000

VALUES OF PRESS. AND VELS. 1TH CYCLE

NODE PRESS N.DIVERGENCE U-VELOCITY U-INCREMENT V-VELOCITY V-INCR

<table>
<thead>
<tr>
<th>NODE</th>
<th>Press</th>
<th>U-Vel</th>
<th>U-Incr</th>
<th>V-Vel</th>
<th>V-Incr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.00E 00</td>
<td>0.24E 01</td>
<td>0.24E 01</td>
<td>-0.28E 01</td>
</tr>
<tr>
<td>2</td>
<td>0.13E 04</td>
<td>0.00E 00</td>
<td>0.60E 00</td>
<td>0.60E 00</td>
<td>0.53E 00</td>
</tr>
<tr>
<td>3</td>
<td>0.17E 05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.12E 05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.10E 05</td>
<td>0.00E 00</td>
<td>0.10E 01</td>
<td>0.10E 01</td>
<td>0.19E 01</td>
</tr>
<tr>
<td>6</td>
<td>0.42E 03</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>-0.27E 04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>0.11E 05</td>
<td>0.00E 00</td>
<td>0.10E 01</td>
<td>0.10E 01</td>
<td>-0.19E 01</td>
</tr>
<tr>
<td>9</td>
<td>0.12E 05</td>
<td>0.00E 00</td>
<td>0.10E 01</td>
<td>0.10E 01</td>
<td>-0.62E 00</td>
</tr>
<tr>
<td>10</td>
<td>-0.41E 05</td>
<td>-0.33E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>-0.41E 05</td>
<td>-0.33E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>0.54E 05</td>
<td>-0.67E 00</td>
<td>0.10E 01</td>
<td>0.10E 01</td>
<td>0.62E 00</td>
</tr>
<tr>
<td>13</td>
<td>0.13E 05</td>
<td>-0.67E 00</td>
<td>0.10E 01</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE   -0.6666667E 00
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE   0.512820E-01
SUM OF NODAL DIVERGENCE   -0.200000E 01
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE   0.153846E 00

VALUES OF PRESS. AND VELS. 2TH CYCLE

NODE PRESS N.DIVERGENCE U-VELOCITY U-INCREMENT V-VELOCITY V-INCR

<table>
<thead>
<tr>
<th>NODE</th>
<th>Press</th>
<th>U-Vel</th>
<th>U-Incr</th>
<th>V-Vel</th>
<th>V-Incr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.12E 01</td>
<td>-0.96E 00</td>
<td>-0.34E 01</td>
<td>0.39E 01</td>
</tr>
<tr>
<td>2</td>
<td>-0.14E 04</td>
<td>-0.84E-06</td>
<td>0.16E 01</td>
<td>0.96E 00</td>
<td>-0.75E 00</td>
</tr>
<tr>
<td>3</td>
<td>-0.41E 05</td>
<td>0.80E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>-0.29E 05</td>
<td>0.80E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>-0.22E 05</td>
<td>-0.11E-05</td>
<td>0.10E 01</td>
<td>0.33E-05</td>
<td>-0.26E 01</td>
</tr>
<tr>
<td>6</td>
<td>0.37E 04</td>
<td>0.35E-06</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>0.11E 05</td>
<td>-0.70E-06</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>-0.21E 05</td>
<td>-0.70E-06</td>
<td>0.10E 01</td>
<td>0.35E-05</td>
<td>0.26E 01</td>
</tr>
<tr>
<td>9</td>
<td>-0.22E 05</td>
<td>-0.87E-06</td>
<td>0.10E 01</td>
<td>0.14E-05</td>
<td>0.87E 00</td>
</tr>
<tr>
<td>10</td>
<td>0.18E 05</td>
<td>-0.10E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.16E 05</td>
<td>-0.18E-06</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>-0.31E 05</td>
<td>-0.11E-05</td>
<td>0.10E 01</td>
<td>0.36E-06</td>
<td>-0.87E 00</td>
</tr>
<tr>
<td>NODE PRESS N.DIVERGENCE</td>
<td>U-VELOCITY U-INCREMENT</td>
<td>V-VELOCITY V-INCREMENT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0.10E 00 -0.17E 01</td>
<td>0.37E 01 0.47E 01 -0.55E 01 -0.94E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 0.20E 04 0.65E-06</td>
<td>0.22E 00 -0.13E 01 0.10E 01 0.18E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 0.58E 05 -0.11E 01</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 0.41E 05 -0.11E 01</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 0.30E 05 0.68E-05</td>
<td>0.10E 01 -0.44E-05 0.37E 01 0.63E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 -0.51E 04 -0.48E-06</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 -0.16E 05 -0.30E-06</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 0.30E 05 0.41E-06</td>
<td>0.10E 01 -0.58E-05 -0.37E 01 -0.63E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 0.30E 05 0.11E-05</td>
<td>0.10E 01 -0.19E-05 -0.12E 01 -0.21E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 -0.11E 06 -0.52E-07</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 -0.10E 06 -0.15E-06</td>
<td>0.00E 00 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 0.12E 06 -0.29E-06</td>
<td>0.10E 01 -0.36E-06 0.12E 01 0.21E 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 0.30E 05 0.16E-06</td>
<td>0.10E 01 0.00E 00 0.00E 00 0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE: 0.933327E 00
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE: 0.447861E 00
SUM OF NODAL DIVERGENCE: 0.279998E 01
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE: 0.215384E 00
VALUES OF PRESS. AND VELS. 5TH CYCLE

<table>
<thead>
<tr>
<th>NODE PRESS</th>
<th>N.DIVERGENCE</th>
<th>U-VELOCITY</th>
<th>U-INCREMENT</th>
<th>V-VELOCITY</th>
<th>V-INCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>-0.33E 01</td>
<td>0.64E 01</td>
<td>0.92E 01</td>
<td>-0.11E 02</td>
</tr>
<tr>
<td>2</td>
<td>0.39E 04</td>
<td>0.99E-06</td>
<td>-0.54E 00</td>
<td>-0.26E 01</td>
<td>0.20E 01</td>
</tr>
<tr>
<td>3</td>
<td>0.11E 06</td>
<td>-0.22E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.80E 05</td>
<td>-0.22E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.59E 05</td>
<td>0.96E-05</td>
<td>0.10E 01</td>
<td>-0.70E-05</td>
<td>0.72E 01</td>
</tr>
<tr>
<td>6</td>
<td>-0.10E 05</td>
<td>-0.12E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>-0.30E 05</td>
<td>-0.14E-09</td>
<td>0.00E 00</td>
<td>-0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>0.59E 05</td>
<td>0.24E-05</td>
<td>0.10E 01</td>
<td>-0.91E-05</td>
<td>0.72E 01</td>
</tr>
<tr>
<td>9</td>
<td>0.59E 05</td>
<td>0.11E-05</td>
<td>0.10E 01</td>
<td>-0.12E-05</td>
<td>0.24E 01</td>
</tr>
<tr>
<td>10</td>
<td>0.20E 05</td>
<td>-0.27E-06</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.26E 05</td>
<td>0.12E-07</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>0.16E 05</td>
<td>-0.45E-06</td>
<td>0.10E 01</td>
<td>-0.12E-05</td>
<td>0.24E 01</td>
</tr>
<tr>
<td>13</td>
<td>0.59E 05</td>
<td>0.16E-06</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE  0.256101E 01
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE  0.122861E 01
SUM OF NODAL DIVERGENCE  -0.768303E 01
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE  0.591004E 00

VALUES OF PRESS. AND VELS. 6TH CYCLE

<table>
<thead>
<tr>
<th>NODE PRESS</th>
<th>N.DIVERGENCE</th>
<th>U-VELOCITY</th>
<th>U-INCREMENT</th>
<th>V-VELOCITY</th>
<th>V-INCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.46E 01</td>
<td>-0.65E 01</td>
<td>-0.13E 02</td>
<td>0.15E 02</td>
</tr>
<tr>
<td>2</td>
<td>-0.55E 04</td>
<td>-0.46E-05</td>
<td>0.32E 01</td>
<td>0.37E 01</td>
<td>-0.29E 01</td>
</tr>
<tr>
<td>3</td>
<td>-0.16E 06</td>
<td>0.31E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.11E 06</td>
<td>0.31E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>-0.83E 05</td>
<td>0.33E-05</td>
<td>0.10E 01</td>
<td>0.13E-05</td>
<td>-0.10E 02</td>
</tr>
<tr>
<td>6</td>
<td>0.14E 05</td>
<td>0.63E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>0.43E 05</td>
<td>-0.42E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>-0.82E 05</td>
<td>-0.84E-06</td>
<td>-0.10E 01</td>
<td>0.18E-04</td>
<td>0.10E 02</td>
</tr>
<tr>
<td>9</td>
<td>-0.83E 05</td>
<td>-0.44E-05</td>
<td>0.10E 01</td>
<td>-0.51E-05</td>
<td>0.33E 01</td>
</tr>
<tr>
<td>10</td>
<td>0.29E 06</td>
<td>-0.23E-06</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.28E 06</td>
<td>-0.12E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>-0.34E 06</td>
<td>-0.92E-06</td>
<td>0.10E 01</td>
<td>0.17E-05</td>
<td>-0.33E 01</td>
</tr>
<tr>
<td>13</td>
<td>-0.83E 05</td>
<td>-0.64E-06</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE  0.358540E 01
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE  0.172024E 01
SUM OF NODAL DIVERGENCE  0.107562E 02
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE  0.827402E 00

VALUES OF PRESS. AND VELS. 7TH CYCLE

<table>
<thead>
<tr>
<th>NODE PRESS</th>
<th>N.DIVERGENCE</th>
<th>U-VELOCITY</th>
<th>U-INCREMENT</th>
<th>V-VELOCITY</th>
<th>V-INCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>-0.65E 01</td>
<td>0.12E 02</td>
<td>0.18E 02</td>
<td>-0.21E 02</td>
</tr>
</tbody>
</table>
### Node Press N. Difference u-Reloc-tion-Icrement a-Reloc-ity a-Inch

#### Values of Press, and Veloc. 4TH Cycle

<table>
<thead>
<tr>
<th>Node</th>
<th>Press</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

#### Values of Press, and Veloc. 8TH Cycle

<table>
<thead>
<tr>
<th>Node</th>
<th>Press</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Node</td>
<td>Press (N)</td>
<td>Divergence</td>
</tr>
<tr>
<td>------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>1</td>
<td>0.10E+00</td>
<td>0.18E+02</td>
</tr>
<tr>
<td>2</td>
<td>-0.21E+05</td>
<td>0.37E+05</td>
</tr>
<tr>
<td>3</td>
<td>-0.61E+06</td>
<td>0.12E+02</td>
</tr>
<tr>
<td>4</td>
<td>-0.43E+06</td>
<td>0.12E+02</td>
</tr>
<tr>
<td>5</td>
<td>-0.32E+06</td>
<td>-0.24E-04</td>
</tr>
<tr>
<td>6</td>
<td>0.54E+05</td>
<td>-0.55E-06</td>
</tr>
<tr>
<td>7</td>
<td>0.16E+06</td>
<td>-0.34E-05</td>
</tr>
<tr>
<td>8</td>
<td>-0.31E+06</td>
<td>0.10E-04</td>
</tr>
<tr>
<td>9</td>
<td>-0.32E+06</td>
<td>-0.18E-04</td>
</tr>
<tr>
<td>10</td>
<td>0.11E+07</td>
<td>0.98E-06</td>
</tr>
<tr>
<td>11</td>
<td>0.11E+07</td>
<td>-0.57E-05</td>
</tr>
<tr>
<td>12</td>
<td>-0.13E+07</td>
<td>-0.36E-05</td>
</tr>
<tr>
<td>13</td>
<td>-0.32E+06</td>
<td>-0.68E-05</td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE: 0.137735E+02
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE: 0.660813E+01
SUM OF NODAL DIVERGENCE: 0.413204E+02
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE: 0.317850E+01
### 12th Cycle

<table>
<thead>
<tr>
<th>Node</th>
<th>Press (N/divergence)</th>
<th>U-Velocity (U-increment)</th>
<th>V-Velocity (V-increment)</th>
<th>V-Incr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.35E 02</td>
<td>-0.56E 02</td>
<td>-0.97E 02</td>
</tr>
<tr>
<td>2</td>
<td>-0.41E 05</td>
<td>-0.28E-04</td>
<td>0.17E 02</td>
<td>0.28E 02</td>
</tr>
<tr>
<td>3</td>
<td>-0.12E 07</td>
<td>0.23E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>-0.84E 06</td>
<td>0.23E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>-0.62E 06</td>
<td>0.40E-04</td>
<td>0.10E 01</td>
<td>0.36E-04</td>
</tr>
<tr>
<td>6</td>
<td>0.10E 06</td>
<td>0.36E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>0.32E 06</td>
<td>-0.25E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>-0.61E 06</td>
<td>-0.60E-05</td>
<td>0.10E 01</td>
<td>0.57E-04</td>
</tr>
<tr>
<td>9</td>
<td>-0.62E 06</td>
<td>-0.17E-04</td>
<td>0.10E 01</td>
<td>0.12E-04</td>
</tr>
<tr>
<td>10</td>
<td>0.36E 06</td>
<td>0.52E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.30E 06</td>
<td>-0.88E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>-0.73E 06</td>
<td>0.75E-06</td>
<td>0.10E 01</td>
<td>0.49E-05</td>
</tr>
<tr>
<td>13</td>
<td>-0.62E 06</td>
<td>-0.55E-05</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

### 13th Cycle

<table>
<thead>
<tr>
<th>Node</th>
<th>Press (N/divergence)</th>
<th>U-Velocity (U-increment)</th>
<th>V-Velocity (V-increment)</th>
<th>V-Incr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>-0.49E 02</td>
<td>0.80E 02</td>
<td>0.14E 03</td>
</tr>
<tr>
<td>2</td>
<td>0.60E 05</td>
<td>0.32E-04</td>
<td>-0.22E 02</td>
<td>-0.39E 02</td>
</tr>
<tr>
<td>3</td>
<td>0.17E 07</td>
<td>-0.32E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.12E 07</td>
<td>-0.32E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.89E 06</td>
<td>0.16E-03</td>
<td>0.10E 01</td>
<td>0.10E-03</td>
</tr>
<tr>
<td>6</td>
<td>-0.15E 06</td>
<td>-0.21E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>-0.45E 06</td>
<td>0.13E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>0.87E 06</td>
<td>0.59E-05</td>
<td>0.10E 01</td>
<td>-0.15E-03</td>
</tr>
<tr>
<td>9</td>
<td>0.89E 06</td>
<td>0.19E-04</td>
<td>0.10E 01</td>
<td>-0.41E-04</td>
</tr>
<tr>
<td>10</td>
<td>-0.31E 07</td>
<td>0.65E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>-0.30E 07</td>
<td>0.16E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>0.36E 07</td>
<td>0.43E-05</td>
<td>0.10E 01</td>
<td>-0.23E-04</td>
</tr>
<tr>
<td>13</td>
<td>0.89E 06</td>
<td>-0.22E-05</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

### 14th Cycle

<table>
<thead>
<tr>
<th>Node</th>
<th>Press (N/divergence)</th>
<th>U-Velocity (U-increment)</th>
<th>V-Velocity (V-increment)</th>
<th>V-Incr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.377E 02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.181E 02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-0.113E 03</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.872E 01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Node</td>
<td>Press</td>
<td>Divergence</td>
<td>u-Velocity</td>
<td>Increment</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>------------</td>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.68E 02</td>
<td>-0.11E 03</td>
<td>-0.19E 03</td>
</tr>
<tr>
<td>2</td>
<td>-0.78E 05</td>
<td>0.95E-06</td>
<td>0.33E 02</td>
<td>0.54E 02</td>
</tr>
<tr>
<td>3</td>
<td>-0.23E 07</td>
<td>0.45E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.16E 07</td>
<td>0.45E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>-0.12E 07</td>
<td>0.40E-04</td>
<td>0.10E 01</td>
<td>0.20E-03</td>
</tr>
<tr>
<td>6</td>
<td>0.20E 06</td>
<td>0.50E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>0.62E 06</td>
<td>0.78E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>-0.12E 07</td>
<td>0.33E-04</td>
<td>0.10E 01</td>
<td>0.18E-03</td>
</tr>
<tr>
<td>9</td>
<td>-0.12E 07</td>
<td>0.67E-07</td>
<td>0.10E 01</td>
<td>0.36E-04</td>
</tr>
<tr>
<td>10</td>
<td>0.62E 06</td>
<td>0.23E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.49E 06</td>
<td>0.37E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>-0.13E 07</td>
<td>0.20E-04</td>
<td>0.10E 01</td>
<td>0.30E-04</td>
</tr>
<tr>
<td>13</td>
<td>-0.12E 07</td>
<td>0.17E-04</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

**SUM OF ELEMENT DIVERGENCE**: 0.529112E 02
**AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE**: 0.253852E 02
**SUM OF NODAL DIVERGENCE**: 0.158733E 03
**AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE**: 0.122103E 02

**VALUES OF PRESS. AND VELS. 15TH CYCLE**

<table>
<thead>
<tr>
<th>Node</th>
<th>Press</th>
<th>Divergence</th>
<th>u-Velocity</th>
<th>Increment</th>
<th>v-Velocity</th>
<th>Increment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.95E 02</td>
<td>0.16E 03</td>
<td>0.27E 03</td>
<td>-0.31E 03</td>
<td>-0.53E 03</td>
</tr>
<tr>
<td>2</td>
<td>0.12E 06</td>
<td>0.44E-04</td>
<td>-0.43E 02</td>
<td>-0.76E 02</td>
<td>0.59E 02</td>
<td>0.10E 03</td>
</tr>
<tr>
<td>3</td>
<td>0.33E 07</td>
<td>-0.63E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.23E 07</td>
<td>-0.63E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.18E 07</td>
<td>0.37E-03</td>
<td>0.10E 01</td>
<td>-0.19E-03</td>
<td>0.21E 03</td>
<td>0.36E 03</td>
</tr>
<tr>
<td>6</td>
<td>-0.30E 06</td>
<td>0.29E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>-0.89E 06</td>
<td>0.34E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>0.17E 07</td>
<td>0.72E-04</td>
<td>0.10E 01</td>
<td>-0.37E-03</td>
<td>-0.21E 03</td>
<td>-0.36E 03</td>
</tr>
<tr>
<td>9</td>
<td>0.18E 07</td>
<td>0.34E-04</td>
<td>0.10E 01</td>
<td>-0.95E-05</td>
<td>-0.69E 02</td>
<td>-0.12E 03</td>
</tr>
<tr>
<td>10</td>
<td>0.12E 07</td>
<td>0.15E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>0.14E 07</td>
<td>0.30E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>-0.18E 06</td>
<td>-0.15E-05</td>
<td>0.10E 01</td>
<td>-0.73E-04</td>
<td>0.69E 02</td>
<td>0.12E 03</td>
</tr>
<tr>
<td>13</td>
<td>0.18E 07</td>
<td>0.29E-05</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

**SUM OF ELEMENT DIVERGENCE**: -0.740753E 02
**AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE**: 0.355389E 02
**SUM OF NODAL DIVERGENCE**: -0.222226E 03
**AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE**: 0.170944E 02

**VALUES OF PRESS. AND VELS. 16TH CYCLE**

<table>
<thead>
<tr>
<th>Node</th>
<th>Press</th>
<th>Divergence</th>
<th>u-Velocity</th>
<th>Increment</th>
<th>v-Velocity</th>
<th>Increment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>0.13E 03</td>
<td>-0.22E 03</td>
<td>-0.37E 03</td>
<td>0.44E 03</td>
<td>0.75E 03</td>
</tr>
<tr>
<td>2</td>
<td>-0.15E 06</td>
<td>-0.45E-04</td>
<td>0.63E 02</td>
<td>0.11E 03</td>
<td>-0.83E 02</td>
<td>-0.14E 03</td>
</tr>
<tr>
<td>3</td>
<td>-0.44E 07</td>
<td>0.89E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>-0.32E 07</td>
<td>0.89E 02</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>7</td>
<td>0.12E 07  -0.13E-03  0.00E 00  0.00E 00  0.00E 00  0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-0.23E 07  -0.38E-04  0.10E 01  0.75E-03  0.29E 03  0.50E 03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-0.23E 07  -0.17E-03  0.10E 01  -0.14E-03  0.97E 02  0.17E 03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.83E 07  0.55E-05  0.00E 00  0.00E 00  0.00E 00  0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.83E 07  -0.14E-04  0.00E 00  0.00E 00  0.00E 00  0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-0.97E 07  -0.45E-05  0.10E 01  0.12E-03  -0.97E 02  -0.17E 03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>-0.23E 07  -0.46E-04  0.10E 01  0.00E 00  0.00E 00  0.00E 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE 0.103705E 03
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE 0.497544E 02
SUM OF NODAL DIVERGENCE 0.311115E 03
AVERAGE ABSOLUTE VALUE OF NODAL DIVERGENCE 0.239320E 02

VALUES OF PRESS. AND VELOS. 17TH CYCLE

<table>
<thead>
<tr>
<th>NODE</th>
<th>PRESS</th>
<th>NODE DIVERGENCE</th>
<th>U VELOCITY</th>
<th>U INCREMENT</th>
<th>V VELOCITY</th>
<th>V INCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10E 00</td>
<td>-0.19E 03</td>
<td>0.31E 03</td>
<td>0.52E 03</td>
<td>-0.61E 03</td>
<td>-0.10E 04</td>
</tr>
<tr>
<td>2</td>
<td>0.25E 06</td>
<td>-0.10E-03</td>
<td>-0.86E 02</td>
<td>-0.15E 03</td>
<td>0.12E 03</td>
<td>0.20E 03</td>
</tr>
<tr>
<td>3</td>
<td>0.67E 07</td>
<td>-0.12E 03</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>4</td>
<td>0.46E 07</td>
<td>-0.12E 03</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>5</td>
<td>0.35E 07</td>
<td>0.43E-03</td>
<td>0.10E 01</td>
<td>-0.24E-03</td>
<td>0.41E 03</td>
<td>0.70E 03</td>
</tr>
<tr>
<td>6</td>
<td>-0.61E 06</td>
<td>0.74E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>7</td>
<td>-0.18E 07</td>
<td>-0.56E-04</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>8</td>
<td>0.34E 07</td>
<td>0.35E-04</td>
<td>0.10E 01</td>
<td>-0.10E-02</td>
<td>-0.41E 03</td>
<td>-0.70E 03</td>
</tr>
<tr>
<td>9</td>
<td>0.35E 07</td>
<td>0.25E-03</td>
<td>0.10E 01</td>
<td>-0.12E-03</td>
<td>-0.14E 03</td>
<td>-0.23E 03</td>
</tr>
<tr>
<td>10</td>
<td>-0.12E 08</td>
<td>-0.20E-05</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>11</td>
<td>-0.11E 08</td>
<td>-0.10E-03</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
<tr>
<td>12</td>
<td>0.14E 08</td>
<td>-0.11E-03</td>
<td>0.10E 01</td>
<td>-0.17E-03</td>
<td>0.14E 03</td>
<td>0.23E 03</td>
</tr>
<tr>
<td>13</td>
<td>0.35E 07</td>
<td>0.37E-04</td>
<td>0.10E 01</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
<td>0.00E 00</td>
</tr>
</tbody>
</table>

SUM OF ELEMENT DIVERGENCE -0.145186E 03
AVERAGE ABSOLUTE VALUE OF ELEMENT DIVERGENCE 0.696557E 02
SUM OF NODAL DIVERGENCE -0.435560E 03
5.4 General Examples of the Use of FEHPOL

These examples deal primarily with the ability of the system to define and manipulate the geometrical entities associated with a user-oriented definition of a finite element problem. The commands required to generate and display different CURVES, REGIONS, and NODES are given and the resultant actions are shown. Example 15 shows the definition of a CURVE using both intermediate CURVES and interpolation.

```
CURVE 1,4 3,4 5,5 6,8 5,9 4,8 4,2 5,1 6,2 5,5:
CURVE 5,5 8,5 9,6 8,7 7,6 7,5 3,4 9,4:
CURVE E(3) 11,6 12,8 11,10 11,4 12,7 13,7 14,4 15,4:
CURVE 15,4 16,7 16,1 15,3 17,7 18,6 18,5 17,4 19,4:
CURVE P(9) 21,7 22,7 23,6 23,5 22,4 21,4 20,5 20,6 23,6:
CURVE 23,6 25,6 25,8 25,10 24,3 25,4 27,4 28,5:
INTERPOLATE F NODES=20,SMOOTH
INTERPOLATE E NODES=18,SMOOTH
INTERPOLATE H NODES=20,SMOOTH
INTERPOLATE P NODES=22,SMOOTH
INTERPOLATE O NODES=19,SMOOTH
INTERPOLATE L NODES=17,SMOOTH
CURVE F E H P O L:FEHPOL
SET XMAX=30,YMAX=10
PLOT FEHPOL
```

Example 15
The following example shows the output format and the pos-
WRITE commands. The forms of the representations of curves and
parameters used by the system are apparent. Note, for instance
the null symbols in the symbol pool which were the CURVES WEST-
BANK and EASTBANK before interpolation. Besides printed output
from the WRITE commands, this example includes several plotting
instructions, with the plots generated by them given in some of
the following pages.

NODE 13,14:IRONWKS
CUR 15.5,25 16,24 16,22 14,21 7,19 6,18 7,17 9,16 10,17 12,17
16,17 19,16 21,14 19,10 15.5,9 10,5 11,3 11,2 11,1:EASTBANK
CURVE 7,1 7,2 6,3 5,4 6,7 12,10 14,11 16,12 15,13 IRONWKS 7,15
5,16 3,17 4,19 7,21 10,21 13,23 14,24 14,25:WESTBANK
INTERPOLATE WESTBANK NODES=28,REGULAR
INTERPOLATE EASTBANK NODES=25,Smooth
CURVE 14,25 EASTBANK(1):COBDEN-BRIDGE
CURVE 11,1 WESTBANK(1):FLOATING-BRIDGE
REGION EASTBANK FLOATING-BRIDGE WESTBANK COBDEN-BRIDGE:ITCHEN
MESH ITCHEN WITH SIX NODED TRIANGLES
REGION EASTBANK(1-6) WESTBANK(14-19):UPPERPART
SET CONCENTRATION=0 ON COBDEN-BRIDGE
SET DISCHARGE=100 ON IRONWKS
SET DEPTH=1 ON ITCHEN,DEPTH=0.5 ON UPPERPART
SET DC=0.1

* ABOVE IS A TYPICAL PROBLEM DESCRIPTION. THE
* COMMANDS TO DISPLAY AND PLOT THE ITEMS WHICH
* HAVE BEEN DEFINED ARE BELOW. IN PRACTICE, THESE
* COMMANDS WOULD BE ISSUED INTERACTIVELY TO SEE
* WHAT DATA HAD BEEN GENERATED BY THE SYSTEM OR
* TO FIND OUT WHERE A MISTAKE IN DEFINITIONS
* HAD BEEN MADE BY THE USER. ON THE NEXT FEW
* PAGES THE OUTPUT FROM THE WRITE AND PLOT COM-
* MANDS BELOW ARE GIVEN.

WRITE FLOATING-BRIDGE,CONNECTIVITY,DEPTH
WRITE SYMBOLS,PARAMETERS
WRITE MESHESPEC,NODAL COORDINATES
SET XMAX=52,YMAX=26
ASSIGN 6 TO SOURCE
PLOT AND LABEL MESH
PLOT EASTBANK
PLOT NODES

Example 16
### MODAL COORDINATES

**CONNECTIVITY TABLE:**

<table>
<thead>
<tr>
<th></th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.10000000E 01</td>
<td>1.0000000E 00</td>
</tr>
<tr>
<td>2</td>
<td>7.00000000E 00</td>
<td>1.0000000E 00</td>
</tr>
</tbody>
</table>

**NUMBER OF SYMBOLS:** 9

**SYMBOL SPACE LEFT:** 120

<table>
<thead>
<tr>
<th></th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>2</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>3</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>4</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>5</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>6</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>7</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
<tr>
<td>8</td>
<td>7.6500</td>
<td>1.1000</td>
</tr>
</tbody>
</table>

**NODAL COORDINATES**

<table>
<thead>
<tr>
<th></th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.5000</td>
<td>25.0000</td>
</tr>
<tr>
<td>2</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>3</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>4</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>5</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>6</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>7</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>8</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>9</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>10</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>11</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>12</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>13</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>14</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>15</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>16</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>17</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>18</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>19</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>20</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>21</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>22</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>23</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>24</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>25</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>26</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
<tr>
<td>27</td>
<td>14.7500</td>
<td>25.0000</td>
</tr>
</tbody>
</table>

... (Continued with more nodal coordinates)
<table>
<thead>
<tr>
<th>PARAMETER TABLE:</th>
<th>N/E</th>
<th>VALUE</th>
<th>N/E</th>
<th>VALUE</th>
<th>N/E</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPTH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.01000E 01</td>
<td>2</td>
<td>0.50000E 00</td>
<td>3</td>
<td>0.50000E 00</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.50000E 00</td>
<td>5</td>
<td>0.50000E 00</td>
<td>6</td>
<td>0.50000E 00</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.50000E 00</td>
<td>8</td>
<td>0.50000E 00</td>
<td>9</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.10000E 01</td>
<td>11</td>
<td>0.10000E 01</td>
<td>12</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.10000E 01</td>
<td>14</td>
<td>0.10000E 01</td>
<td>15</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.10000E 01</td>
<td>17</td>
<td>0.10000E 01</td>
<td>18</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.10000E 01</td>
<td>20</td>
<td>0.50000E 00</td>
<td>21</td>
<td>0.50000E 00</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>0.50000E 00</td>
<td>23</td>
<td>0.50000E 00</td>
<td>24</td>
<td>0.50000E 00</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>0.50000E 00</td>
<td>26</td>
<td>0.50000E 00</td>
<td>27</td>
<td>0.50000E 00</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0.10000E 01</td>
<td>29</td>
<td>0.10000E 01</td>
<td>30</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>0.10000E 01</td>
<td>32</td>
<td>0.10000E 01</td>
<td>33</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>0.10000E 01</td>
<td>35</td>
<td>0.10000E 01</td>
<td>36</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>0.10000E 01</td>
<td>38</td>
<td>0.10000E 01</td>
<td>39</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>0.10000E 01</td>
<td>41</td>
<td>0.10000E 01</td>
<td>42</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>0.10000E 01</td>
<td>44</td>
<td>0.10000E 01</td>
<td>45</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>0.10000E 01</td>
<td>47</td>
<td>0.10000E 01</td>
<td>48</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>0.10000E 01</td>
<td>50</td>
<td>0.10000E 01</td>
<td>51</td>
<td>0.10000E 01</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>0.10000E 01</td>
<td>53</td>
<td>0.10000E 01</td>
<td>54</td>
<td>0.10000E 01</td>
<td></td>
</tr>
</tbody>
</table>

| NUMBER OF PARAMETERS= | 4 |
| PARAMETER SPACE LEFT= | 595 |

<table>
<thead>
<tr>
<th>SYMBOL TABLE:</th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRONWES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.3000000E 01</td>
<td>1.4000000E 01</td>
</tr>
</tbody>
</table>

<p>| 1             | 1.5500000E 01 | 2.5000000E 01 |
| 2             | 1.6000000E 01 | 2.4000000E 01 |
| 3             | 1.6000000E 01 | 2.2000000E 01 |
| 4             | 1.4000000E 01 | 2.1000000E 01 |
| 5             | 7.0000000E 00 | 1.9000000E 01 |
| 6             | 5.0000000E 00 | 1.8000000E 01 |
| 7             | 7.0000000E 00 | 1.7000000E 01 |</p>
<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>9.0000000E 00</th>
<th>1.6000000E 01</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1.0000000E 01</td>
<td>1.7000000E 01</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.2000000E 01</td>
<td>1.7000000E 01</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1.6000000E 01</td>
<td>1.6000000E 01</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2.1000000E 01</td>
<td>1.4000000E 01</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1.0000000E 01</td>
<td>1.0000000E 01</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1.5500000E 01</td>
<td>9.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1.0000000E 01</td>
<td>5.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.1000000E 01</td>
<td>3.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1.1000000E 01</td>
<td>2.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1.1000000E 01</td>
<td>1.0000000E 00</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>7.0000000E 00</th>
<th>1.0000000E 00</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>7.0000000E 00</td>
<td>2.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6.0000000E 00</td>
<td>3.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5.0000000E 00</td>
<td>4.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6.0000000E 00</td>
<td>7.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.2000000E 01</td>
<td>1.0000000E 01</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.4000000E 01</td>
<td>1.1000000E 01</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.5000000E 01</td>
<td>1.2000000E 01</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1.3000000E 01</td>
<td>1.3000000E 01</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.3000000E 01</td>
<td>1.4000000E 01</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>7.0000000E 00</td>
<td>1.5000000E 01</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>5.0000000E 00</td>
<td>1.6000000E 01</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>3.0000000E 00</td>
<td>1.7000000E 01</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>4.0000000E 00</td>
<td>1.9000000E 01</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>7.0000000E 00</td>
<td>2.1000000E 01</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.0000000E 01</td>
<td>2.1500000E 01</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1.3000000E 01</td>
<td>2.3000000E 01</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1.4000000E 01</td>
<td>2.4000000E 01</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1.4000000E 01</td>
<td>2.5000000E 01</td>
<td></td>
</tr>
</tbody>
</table>

&WESTBANK

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>7.0000000E 00</th>
<th>1.0000000E 00</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>7.0000000E 00</td>
<td>2.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6.0000000E 00</td>
<td>3.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5.0000000E 00</td>
<td>4.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5.0625000E 00</td>
<td>5.3750000E 00</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6.0000000E 00</td>
<td>7.0000000E 00</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8.0375000E 00</td>
<td>8.6250000E 00</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.2000000E 01</td>
<td>1.0000000E 01</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1.2500000E 01</td>
<td>1.0625000E 01</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.4000000E 01</td>
<td>1.1000000E 01</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1.5187500E 01</td>
<td>1.1562500E 01</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1.6000000E 01</td>
<td>1.2000000E 01</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1.5000000E 01</td>
<td>1.3000000E 01</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1.4375000E 01</td>
<td>1.2500000E 01</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1.3000000E 01</td>
<td>1.4000000E 01</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.0000000E 01</td>
<td>1.4500000E 01</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>7.0000000E 00</td>
<td>1.5000000E 01</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>5.749999999E 00</td>
<td>1.5500000E 01</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>5.0000000E 00</td>
<td>1.6000000E 01</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>3.8125000E 00</td>
<td>1.6437500E 01</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>1.0000000E 00</td>
<td>1.7000000E 01</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>3.1875000E 00</td>
<td>1.7975000E 01</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>4.0000000E 00</td>
<td>1.9000000E 01</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>5.3750000E 00</td>
<td>2.000375000E 01</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>7.0000000E 00</td>
<td>2.1000000E 01</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>8.5000000E 00</td>
<td>2.128125000E 01</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1.0000000E 01</td>
<td>2.1500000E 01</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1.1562500E 01</td>
<td>2.221875000E 01</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1.3000000E 01</td>
<td>2.3000000E 01</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1.4000000E 01</td>
<td>2.4000000E 01</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>1.4000000E 01</td>
<td>2.5000000E 01</td>
<td></td>
</tr>
</tbody>
</table>

**&FATFANK**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5500000E 01</td>
<td>2.5000000E 01</td>
</tr>
<tr>
<td>2</td>
<td>1.586739159E 01</td>
<td>2.438079834E 01</td>
</tr>
<tr>
<td>3</td>
<td>1.615030289E 01</td>
<td>2.312492752E 01</td>
</tr>
<tr>
<td>4</td>
<td>1.5892353062E 01</td>
<td>2.182476807E 01</td>
</tr>
<tr>
<td>5</td>
<td>1.444486427E 01</td>
<td>2.113337708E 01</td>
</tr>
<tr>
<td>6</td>
<td>9.695997238E 00</td>
<td>1.979199982E 01</td>
</tr>
<tr>
<td>7</td>
<td>6.2186880116E 00</td>
<td>1.861907196E 01</td>
</tr>
<tr>
<td>8</td>
<td>6.008255959E 00</td>
<td>1.795999085E 01</td>
</tr>
<tr>
<td>9</td>
<td>6.631104469E 00</td>
<td>1.724000168E 01</td>
</tr>
<tr>
<td>10</td>
<td>7.058334923E 00</td>
<td>1.639686584E 01</td>
</tr>
<tr>
<td>11</td>
<td>9.2150015115E 00</td>
<td>1.613600159E 01</td>
</tr>
<tr>
<td>12</td>
<td>9.096965769E 00</td>
<td>1.601757645E 01</td>
</tr>
<tr>
<td>13</td>
<td>1.110182190E 01</td>
<td>1.70522321B1E 01</td>
</tr>
<tr>
<td>14</td>
<td>1.336527388E 01</td>
<td>1.70522321B1E 01</td>
</tr>
<tr>
<td>15</td>
<td>1.627670825E 01</td>
<td>1.695678955E 01</td>
</tr>
<tr>
<td>16</td>
<td>1.847999954E 01</td>
<td>1.628000259E 01</td>
</tr>
<tr>
<td>17</td>
<td>2.035449606E 01</td>
<td>1.514803123E 01</td>
</tr>
<tr>
<td>18</td>
<td>2.079056168E 01</td>
<td>1.30392029E 01</td>
</tr>
<tr>
<td>19</td>
<td>1.912544250E 01</td>
<td>1.013563065E 01</td>
</tr>
<tr>
<td>20</td>
<td>1.581366348E 01</td>
<td>9.359167095E 00</td>
</tr>
<tr>
<td>21</td>
<td>1.306400108E 01</td>
<td>7.480001450E 00</td>
</tr>
<tr>
<td>22</td>
<td>9.242640656E 00</td>
<td>4.674114227E 00</td>
</tr>
<tr>
<td>23</td>
<td>1.071232033E 01</td>
<td>3.226815071E 00</td>
</tr>
<tr>
<td>24</td>
<td>1.105913544E 01</td>
<td>2.380862713E 00</td>
</tr>
<tr>
<td>25</td>
<td>1.1000000E 01</td>
<td>1.0000000E 00</td>
</tr>
</tbody>
</table>

**&CORDEN-BRIDGE**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.4000000E 01</td>
<td>2.5000000E 01</td>
</tr>
<tr>
<td>2</td>
<td>1.5500000E 01</td>
<td>2.5000000E 01</td>
</tr>
</tbody>
</table>

**&FLOATING-BRIDGE**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.1000000E 01</td>
<td>1.0000000E 00</td>
</tr>
<tr>
<td>2</td>
<td>7.0000000E 00</td>
<td>1.0000000E 00</td>
</tr>
</tbody>
</table>

**&KITCHEN**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5500000E 01</td>
<td>2.5000000E 01</td>
</tr>
<tr>
<td>2</td>
<td>1.586739159E 01</td>
<td>2.438079834E 01</td>
</tr>
<tr>
<td>3</td>
<td>1.615030289E 01</td>
<td>2.312492752E 01</td>
</tr>
<tr>
<td>4</td>
<td>1.5892353062E 01</td>
<td>2.182476807E 01</td>
</tr>
<tr>
<td>5</td>
<td>1.444486427E 01</td>
<td>2.113337708E 01</td>
</tr>
<tr>
<td></td>
<td>6.9597238E00</td>
<td>9.7919982E01</td>
</tr>
<tr>
<td>---</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>7</td>
<td>6.21868501E00</td>
<td>1.86190719E01</td>
</tr>
<tr>
<td>8</td>
<td>6.00825595E00</td>
<td>1.79599908E01</td>
</tr>
<tr>
<td>9</td>
<td>6.3110446E00</td>
<td>1.72400168E01</td>
</tr>
<tr>
<td>10</td>
<td>7.9588423E00</td>
<td>1.63968658E01</td>
</tr>
<tr>
<td>11</td>
<td>9.21601511E00</td>
<td>1.613600159E01</td>
</tr>
<tr>
<td>12</td>
<td>9.00060679E00</td>
<td>1.61075704E01</td>
</tr>
<tr>
<td>13</td>
<td>1.11018219E00</td>
<td>1.70522232E01</td>
</tr>
<tr>
<td>14</td>
<td>1.23667288E01</td>
<td>1.70522232E01</td>
</tr>
<tr>
<td>15</td>
<td>1.62767926E01</td>
<td>1.69567895E01</td>
</tr>
<tr>
<td>16</td>
<td>1.84799054E01</td>
<td>1.628000259E01</td>
</tr>
<tr>
<td>17</td>
<td>2.03544600E01</td>
<td>1.514803123E01</td>
</tr>
<tr>
<td>18</td>
<td>2.079056168E01</td>
<td>1.30392029E01</td>
</tr>
<tr>
<td>19</td>
<td>1.91254459E01</td>
<td>1.013568306E01</td>
</tr>
<tr>
<td>20</td>
<td>1.68136634E01</td>
<td>0.9359167099E00</td>
</tr>
<tr>
<td>21</td>
<td>1.36480010E01</td>
<td>7.480001450E00</td>
</tr>
<tr>
<td>22</td>
<td>9.90590566E00</td>
<td>6.674114227E00</td>
</tr>
<tr>
<td>23</td>
<td>1.071232023E01</td>
<td>3.226815701E00</td>
</tr>
<tr>
<td>24</td>
<td>1.16519144E01</td>
<td>2.380862713E00</td>
</tr>
</tbody>
</table>

**UPPER PART**

<table>
<thead>
<tr>
<th></th>
<th>1.5500000E01</th>
<th>2.5000000E01</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.586739159E01</td>
<td>2.428079834E01</td>
</tr>
<tr>
<td>3</td>
<td>1.615030289E01</td>
<td>2.312492752E01</td>
</tr>
</tbody>
</table>
NUMBER OF SYMBOLS = 30
SYMBOL SPACE LEFT = 120

PARAMETER TABLE: N/E VALUE N/E VALUE N/E VALUE

CONCENTRATION
1 0.0000E 00 2 0.0000E 00 5 0.0000E 00

DISCHARGE
88 0.1000E 03

DEPTH
1 0.1000E 01 2 0.5000E 00 3 0.5000E 00
4 0.5000E 00 5 0.5000E 00 6 0.5000E 00
7 0.5000E 00 8 0.5000E 00 9 0.1000E 01
10 0.1000E 01 11 0.1000E 01 12 0.1000E 01
13 0.1000E 01 14 0.1000E 01 15 0.1000E 01
16 0.1000E 01 17 0.1000E 01 18 0.1000E 01
19 0.1000E 01 20 0.5000E 00 21 0.5000E 00
22 0.5000E 00 23 0.5000E 00 24 0.5000E 00
25 0.5000E 00 26 0.5000E 00 27 0.5000E 00
28 0.1000E 01 29 0.1000E 01 30 0.1000E 01
31 0.1000E 01 32 0.1000E 01 33 0.1000E 01
34 0.1000E 01 35 0.1000E 01 36 0.1000E 01
37 0.1000E 01 38 0.1000E 01 39 0.1000E 01
40 0.1000E 01 41 0.1000E 01 42 0.1000E 01
43 0.1000E 01 44 0.1000E 01 45 0.1000E 01
46 0.1000E 01 47 0.1000E 01 48 0.1000E 01
49 0.1000E 01 50 0.1000E 01 51 0.1000E 01
52 0.1000E 01 53 0.1000E 01 54 0.1000E 01

DC
1 0.1000E 00

NUMBER OF PARAMETERS = 4
PARAMETER SPACE LEFT = 595

NUMBER OF ELEMENTS = 54
NUMBER OF CORNER PTS = 56
NUMBER OF NODES = 165
BANDWIDTH = 20
<table>
<thead>
<tr>
<th>NODE</th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.5000</td>
<td>25.0000</td>
</tr>
<tr>
<td>2</td>
<td>14.7500</td>
<td>24.5000</td>
</tr>
<tr>
<td>3</td>
<td>14.0000</td>
<td>25.0000</td>
</tr>
<tr>
<td>4</td>
<td>14.0000</td>
<td>24.0000</td>
</tr>
<tr>
<td>5</td>
<td>14.0000</td>
<td>23.5625</td>
</tr>
<tr>
<td>6</td>
<td>14.2224</td>
<td>22.5667</td>
</tr>
<tr>
<td>7</td>
<td>14.8674</td>
<td>24.3808</td>
</tr>
<tr>
<td>8</td>
<td>16.1503</td>
<td>23.1249</td>
</tr>
<tr>
<td>9</td>
<td>15.8924</td>
<td>21.8248</td>
</tr>
<tr>
<td>10</td>
<td>14.4440</td>
<td>21.1374</td>
</tr>
<tr>
<td>11</td>
<td>13.0340</td>
<td>21.6761</td>
</tr>
<tr>
<td>12</td>
<td>13.0000</td>
<td>23.0000</td>
</tr>
<tr>
<td>13</td>
<td>11.6250</td>
<td>22.2188</td>
</tr>
<tr>
<td>14</td>
<td>10.8125</td>
<td>21.8594</td>
</tr>
<tr>
<td>15</td>
<td>9.8480</td>
<td>20.6460</td>
</tr>
<tr>
<td>16</td>
<td>8.2480</td>
<td>20.3960</td>
</tr>
<tr>
<td>17</td>
<td>10.0000</td>
<td>21.5000</td>
</tr>
<tr>
<td>18</td>
<td>8.5000</td>
<td>21.2813</td>
</tr>
<tr>
<td>19</td>
<td>7.0000</td>
<td>21.0000</td>
</tr>
<tr>
<td>20</td>
<td>6.1875</td>
<td>20.5469</td>
</tr>
<tr>
<td>21</td>
<td>5.7968</td>
<td>19.3664</td>
</tr>
<tr>
<td>22</td>
<td>6.1135</td>
<td>18.2895</td>
</tr>
<tr>
<td>23</td>
<td>4.8875</td>
<td>19.5469</td>
</tr>
<tr>
<td>24</td>
<td>5.0041</td>
<td>18.4800</td>
</tr>
<tr>
<td>25</td>
<td>6.0083</td>
<td>17.9600</td>
</tr>
<tr>
<td>26</td>
<td>4.9104</td>
<td>17.1987</td>
</tr>
<tr>
<td>27</td>
<td>6.3197</td>
<td>17.6000</td>
</tr>
<tr>
<td>28</td>
<td>3.5000</td>
<td>17.1875</td>
</tr>
<tr>
<td>29</td>
<td>3.8125</td>
<td>16.4375</td>
</tr>
<tr>
<td>30</td>
<td>3.4062</td>
<td>16.7188</td>
</tr>
<tr>
<td>31</td>
<td>5.9156</td>
<td>16.6200</td>
</tr>
<tr>
<td>32</td>
<td>6.6311</td>
<td>17.2400</td>
</tr>
<tr>
<td>33</td>
<td>6.8356</td>
<td>16.1200</td>
</tr>
<tr>
<td>34</td>
<td>3.0000</td>
<td>17.0000</td>
</tr>
<tr>
<td>35</td>
<td>6.3750</td>
<td>15.2500</td>
</tr>
<tr>
<td>36</td>
<td>7.4792</td>
<td>15.6984</td>
</tr>
<tr>
<td>37</td>
<td>8.5000</td>
<td>14.7500</td>
</tr>
<tr>
<td>38</td>
<td>8.5872</td>
<td>16.2664</td>
</tr>
<tr>
<td>39</td>
<td>9.0680</td>
<td>15.3180</td>
</tr>
<tr>
<td>40</td>
<td>10.0000</td>
<td>14.5000</td>
</tr>
<tr>
<td>41</td>
<td>10.5500</td>
<td>15.7761</td>
</tr>
<tr>
<td>42</td>
<td>9.0907</td>
<td>16.9171</td>
</tr>
<tr>
<td>43</td>
<td>11.1018</td>
<td>17.0522</td>
</tr>
<tr>
<td>44</td>
<td>12.2340</td>
<td>17.0522</td>
</tr>
<tr>
<td>45</td>
<td>13.1831</td>
<td>15.5261</td>
</tr>
<tr>
<td>46</td>
<td>13.3663</td>
<td>15.0722</td>
</tr>
<tr>
<td>47</td>
<td>14.8215</td>
<td>17.0045</td>
</tr>
<tr>
<td>48</td>
<td>15.2946</td>
<td>15.2284</td>
</tr>
<tr>
<td>49</td>
<td>16.2763</td>
<td>16.9568</td>
</tr>
<tr>
<td>50</td>
<td>17.3784</td>
<td>16.6184</td>
</tr>
<tr>
<td>51</td>
<td>16.7400</td>
<td>14.6400</td>
</tr>
<tr>
<td>52</td>
<td>18.4800</td>
<td>16.2800</td>
</tr>
<tr>
<td>53</td>
<td>19.4172</td>
<td>15.7140</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>109</td>
<td>17.5627</td>
<td>11.0678</td>
</tr>
<tr>
<td>111</td>
<td>15.5038</td>
<td>11.7500</td>
</tr>
<tr>
<td>117</td>
<td>17.9696</td>
<td>9.7474</td>
</tr>
<tr>
<td>119</td>
<td>16.0005</td>
<td>10.4295</td>
</tr>
<tr>
<td>121</td>
<td>14.9388</td>
<td>8.4106</td>
</tr>
<tr>
<td>123</td>
<td>14.5938</td>
<td>11.2500</td>
</tr>
<tr>
<td>129</td>
<td>12.5220</td>
<td>8.7400</td>
</tr>
<tr>
<td>131</td>
<td>11.4043</td>
<td>6.0771</td>
</tr>
<tr>
<td>133</td>
<td>12.6250</td>
<td>10.3125</td>
</tr>
<tr>
<td>135</td>
<td>10.4688</td>
<td>9.3125</td>
</tr>
<tr>
<td>137</td>
<td>9.4311</td>
<td>6.6496</td>
</tr>
<tr>
<td>139</td>
<td>9.0246</td>
<td>4.6741</td>
</tr>
<tr>
<td>141</td>
<td>7.4936</td>
<td>5.0246</td>
</tr>
<tr>
<td>143</td>
<td>8.4623</td>
<td>3.3771</td>
</tr>
<tr>
<td>145</td>
<td>6.0000</td>
<td>7.0000</td>
</tr>
<tr>
<td>147</td>
<td>5.0625</td>
<td>5.3750</td>
</tr>
<tr>
<td>149</td>
<td>5.0712</td>
<td>4.6875</td>
</tr>
<tr>
<td>151</td>
<td>6.5000</td>
<td>2.5000</td>
</tr>
<tr>
<td>153</td>
<td>7.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>155</td>
<td>9.0296</td>
<td>2.1904</td>
</tr>
<tr>
<td>157</td>
<td>7.0000</td>
<td>1.5000</td>
</tr>
<tr>
<td>159</td>
<td>10.8857</td>
<td>2.8038</td>
</tr>
<tr>
<td>161</td>
<td>11.0591</td>
<td>2.3809</td>
</tr>
<tr>
<td>163</td>
<td>11.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>165</td>
<td>7.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

NUMBER OF ELEMENTS= 54
NUMBER OF CORNER PTS= 56
NUMBER OF NODES= 165
The following is an example of REGULAR and SMOOTH interpolation.

Example 17
Example 17 (cont'd)
The steps required to generate a finite element mesh on a typically irregular region are as follows. Firstly, the user defines the boundaries of the region using as few points as he feels necessary for an adequate description. Next, the INTERPOLATE statement is used on the components of the boundary to refine the description and generate more nodes. Finally, the internal nodes are generated using the GENERATE command and a mesh generated. Subsequently, the user may add internal nodes using the LET command if the mesh is ill-formed or he wishes a higher density of internal nodes in a certain area. The next few pages show a typical sequence for generating a mesh on the Solent. The FEHPOL source is given below.

```plaintext
REGION 2.4,1.4 1.6,2.4 2.4,3 2.8,4 5.8,4.6 6,5.2 7.4,5.2 8.2,6.4 7.6,7.4 6.9 6.6,9.8 10.4,6.6 12.8,4.8 13.4,5.4 14,4.8 16,5.2 15.8,2 14.6,1 13.6,2.4 12.4,3 11.2,3 9.4,4.4 8.4,4.4 6.2,2.8 5.4,2.8 4.2 2.8,2:SOLENT
SET XMAX=26,YMAX=10
PLOT SOLENT
INTERPOLATE SOLENT NODES=50,LINEAR
PLOT SOLENT
GENERATE INTERNAL NODES FOR SOLENT
MESH SOLENT WITH THREE NODED TRIANGLES,SMOOTH
PLOT MESH
/`
```

Example 18
Southampton

Fareham

Lymington

Newport

Isle of Wight

--- 5m depth contour

FIG. 10 THE SOLENT SYSTEM
Step 1: Define the Region Boundary
Step 2: Interpolate the Region Boundary
Step 3: Generate the Mesh
6.1 Conclusions and Discussion

The FEHPOL System has proven its viability and utility by supervising the solution of a wide variety of practical problems in the short time since it has been fully operational. In December of 1975 it was utilized by the engineers and scientists who attended the Introductory Course on Finite Elements for Fluid Flow at the University of Southampton with quite favorable results. In the relatively short space of two afternoons not only were the participants able to learn the language, but with modest supervision they were able to solve several finite element problems ranging from seepage under a dam to potential flow in an irregularly shaped region.

The main limitations in the use of the system at present is the scarcity of freely available finite element programs for the solution of other differential equations than those mentioned in Chapter Five, in particular solution programs using disc solving techniques for minicomputer applications. Though reliable codes have been written, many are proprietary programs of various consulting engineering firms and software houses. It is hoped that the more widespread use of FEHPOL will provoke some interest in mutually advantageous sharing of software and add to the library of existing FEHPOL solution codes.

Because it is easy to describe, modify, and solve a problem using the system, testing the validity of a finite element discretization is not only made considerably easier, but also less subject to misinterpretation since more tests can be effected in the same length of time. In particular, since altering the number and distribution of elements and nodes for a given region is simply a matter of a few FEHPOL commands, solving the same prob-
lem with a coarser or finer finite element mesh can be done with a minimum of difficulty, and the solutions compared interactive-
ly using the graphic capabilities of the system.

6.2 Enhancements and Further Developments

Several enhancements to the system are planned for the near future. Extension of the symbol pool to become disc resident if full would allow larger problems and even smaller core requirements to be realized. Although neither of the solution programs presently being used with the FEHPOL System allows mixed finite element types, altering the system to support this is an obvious extension.

A more important extension which would be relatively easy to code, seeing as the FEHPOL System is written almost entirely in reentrant code, is to provide multiuser capability. This can be done by using a time slicing interrupt vectored to an assembly language routine which performed register and status saves and rolled the FEHPOL System’s COMMON data area.

Vector plotting capabilities would augment the present contour plotting ability, particularly for flow problems. In addition a macro definition facility would be easy to implement and would be particularly useful in solution of problems involving a high degree of repetition. Another suggestion is to include the various disc solvers (for banded system of equations) which some finite element programs use as FEHPOL service routines, callable like any other of the supervisor calls. This may be implemented in a version of the system, but not in the basic system since in many cases the ‘solution’ program may simply be generating batch data for another program.
FOOTNOTES
(1) Several codes are being developed at the University of Southampton in the Computational Hydraulics Group. The program FPWAVEL3 being developed is for modelling the interaction of arbitrary waves on offshore structures such as oil drilling platforms. A multipollutant, time dependent dispersion model similar to but more powerful than the program DSPSTD described in Chapter 5 is being developed, also.

(2) Among the well known large analysis systems presently in use are NASTRAN, PAFEC, SAP, GENESYS, CLEAT, SPAN, ICES, BERSAFE, ASAS, ASKA.

(3) A good example of a small interactive system similar to (but less powerful than) the FEHPOL System is the finite element idealization and mesh generation system developed by S. Singh (Reference 16).

(4) These are of course only a subset of the requirements of an interactive computer aided design system. For a more detailed analysis consult references 3 and 7.

(5) Rather than relying on similarities in operating systems to aid machine hardware configuration independence, the FORTRAN language is used to overcome differences. For instance, the word length (in bytes) can be calculated and used in the program by use of FORTRAN, (see segment FPSSIM in Appendix A). By careful programming and a sufficient understanding of the compilation process, such differences as calling by value and calling by address may be eliminated. At present, however, there are two abnormal func-
tions in the system (FPLETR and FPDIGT) soon to be changed, as many minicomputers have marginal FORTRAN IV, such as the Texas on which much of the system development was carried out, it is important to use only the most basic of FORTRAN operations and often circuitousness to keep it general.

(6) The syntax of the language is described in Appendix B, and in Backus Naur Form in Appendix A.

(7) The first word in each of the command types is an imperative verb. The verbs at present defined are ASSIGN, CURVE, DELETE, EXECUTE, GENERATE, INTERPOLATE, KEEP, LET, MESH, NODES, PLOT, QUIT, REGION, SET, TRACE, UNKEEP, and WRITE.


(9) See Reference 6, pages 52-55.

(10) For detailed analysis of syntactic error recovery see Reference 6, pages 320-326

(11) See Reference 4.

(12) See Reference 20.

(13) In the event of extremely wide variation of node density on the boundary and REGIONS whose width is of the order of the smaller internodal distances, it is possible to generate a very large number of nodes within a cell according to the given formula. In practice the number within a cell is made
equal to a maximum allowed value whenever it is exceeded.


10. Nelson, J.M. "Introduction to a System for Hydrodynamic Finite Element Modelling", Notes of the Seminar on Computational Models in Coastal and Harbour Engineering, held at the University of Southampton (July 1975)


APPENDIX A

SOURCE LISTING OF THE FEHPOL SYSTEM
FEHPOL I: FINITE ELEMENT SYSTEM FOR HYDRAULIC ENGINEERING
CURRENT ISSUE DATE: 31MAR76
PART NUMBER: FPLO1
AUTHOR: JAMES M. NELSON
SEGMENT: OVLYIA

FEHPOL IS A COMPUTER SYSTEM COMMAND LANGUAGE WHICH THIS PROGRAM INTERPRETS TO ALLOW USERS TO INITIATE AND COORDINATE THE SOLUTIONS OF PARTIAL DIFFERENTIAL EQUATIONS ON ARBITRARILY SHAPED TWO DIMENSIONAL REGIONS BY THE FINITE ELEMENT TECHNIQUE. THE SYSTEM WAS DEVELOPED AT THE UNIVERSITY OF SOUTHAMPTON FUNDED UNDER U.S.P.L. 87-256, THE FULBRIGHT-HAYES ACT. CORRESPONDENCE REGARDING THE SYSTEM SHOULD BE DIRECTED TO EITHER:

DEPT. OF CIVIL ENGINEERING
THE UNIVERSITY
HIGHFIELD, SOUTHAMPTON UK.

OR
J. M. NELSON
C/O 12461 S.E. 280TH
KENT, WASHINGTON 98031 USA

OR
DR. R. A. ADEY
COMPUTATIONAL MECHANICS LTD.
18 SPRING CRESCENT
PORTSWOOD, SOUTHAMPTON UK

CALL FPSTOP(0)
END

**

SUBROUTINE FPSTOP(I)

SEGMENT NAME: FPSTOP
SEGMENT TYPE: SUBROUTINE
SEGMENT LENGTH IN CARDS: 7
DATE OF LAST ALTERATION: 20/07/76
NUMBER OF LOCAL INTEGERS: 0
NUMBER OF LOCAL REALS: 0
FEHPOL ROUTINES CALLED: FPSSTM
FORTRAN LIBRARY ROUTINES CALLED: NONE
OTHER LIBRARY ROUTINES CALLED: NONE
CALLING ROUTINES: MAIN
COMMONS REFERENCED: PARAMS

REMARKS:
THIS ROUTINE PROCESSES THE 'RETURN TO FEHPOL SYSTEM' SUPERVISOR CALL. WHEN ITS ARGUMENT IS POSITIVE AN
ABORT MESSAGE IS DISPLAYED ON THE REPLY DEVICE WITH
THE ABORT CODE = THE ARGUMENT. A CALL TO IT REPLACES
THE FORTRAN 'STOP I' STATEMENT IN PROGRAMS RUNNING
UNDER FEHPOL SUPERVISION.

TABLE OF ABORT CODES FOR THE FEHPOL SYSTEM 15/08/76

<table>
<thead>
<tr>
<th>ABORT NO.</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-28</td>
<td>RESERVED FOR SOLUTION PROGRAMS</td>
</tr>
<tr>
<td>29</td>
<td>SYMBOL OR PARAMETER POOL OVERFLOW</td>
</tr>
<tr>
<td>30</td>
<td>PARAMETER POOL CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>31</td>
<td>SYMBOL POOL CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>32</td>
<td>MORE THAN 100 SYNTAX ERRORS SINCE INIT/QUIT</td>
</tr>
<tr>
<td>33</td>
<td>ILLEGAL INTERPOLATION CODE</td>
</tr>
<tr>
<td>34</td>
<td>ILLEGAL ELEMENT CODE</td>
</tr>
<tr>
<td>35</td>
<td>NEGATIVE NUMBER OF INTERNAL NODES</td>
</tr>
<tr>
<td>36</td>
<td>TOO MANY EQUALLY ATTRACTIVE CONNECTIONS AvAIL</td>
</tr>
<tr>
<td>37</td>
<td>TOO MANYFINITE ELEMENT NODES FOR SYSTEM SIZE</td>
</tr>
<tr>
<td>38</td>
<td>RENUMBERING TABLE CORRUPTION HAS OCCURRED</td>
</tr>
<tr>
<td>39</td>
<td>TOO MANY FINITE ELEMENT CORNER NODES</td>
</tr>
<tr>
<td>40-45</td>
<td>RESERVED FOR SYSTEM DEVELOPMENT</td>
</tr>
<tr>
<td>46</td>
<td>NOT ENOUGH POOL SPACE LEFT FOR INTERPOLATE</td>
</tr>
<tr>
<td>47-52</td>
<td>RESERVED FOR SYSTEM DEVELOPMENT</td>
</tr>
<tr>
<td>53</td>
<td>ATTEMPTED CONTOUR PLOT WITH UNAVAILABLE FIN. EL</td>
</tr>
<tr>
<td>54</td>
<td>ATTEMPTED CONTOUR PLOT OF NON-NODAL PARAMETER</td>
</tr>
</tbody>
</table>

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
IF(I.GT.0)WRITE(RP,1)I
CALL FPSSTM
1 FORMAT(/5(1H*),14HABORTED, CODE=,I3/) END

FUNCTION FPEXST(N)

C SEGMENT NAME........................FPEXST
C SEGMENT TYPE........................FUNCTION
C SEGMENT LENGTH IN CARDS...............8
C DATE OF LAST ALTERATION..............20/07/76
C NUMBER OF LOCAL INTEGERS.............0
C NUMBER OF LOCAL REALS................0
C FEHPOL ROUTINES CALLED................NEXTIS
C FORTRAN LIBRARY ROUTINES CALLED.......NONE
C OTHER LIBRARY ROUTINES CALLED.........USRPRG
C CALLING ROUTINES......................FPCMND
C COMMONS REFERENCED....................NONE
C REMARKS:
THIS ROUTINE SCANS FOR THE 'EXECUTE' STATEMENT AND ON FINDING IT, INITIATES THE SUPERVISED PROGRAM.

IF(NEXTIS(8HEXECUTE ,7))700,10,900
  10 CALL USRPRG
  700 FPEXST=-1.0
  RETURN
  900 FPEXST=1.0
  RETURN
END

**
FUNCTION FPELPCY(CONEL,ELNUM)
**

INTEGER CONEL(6),ELNUM
INTEGER CY(6,395),NL(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
IF(IE.NE.3.AND.IE.NE.6)STOP 43
  DO 10 I=1,IE
  10 CONEL(I)=CY(I,ELNUM)
RETURN
END

**
FUNCTION FPNCDS(M,N,XX,YY)
**

INTEGER CY(6,395),NL(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
IF(IE.NE.3.AND.IE.NE.6)STOP 43
DO 10 I=1,IE
  10 CONEL(I)=CY(I,ELNUM)
RETURN
END
FORTRAN LIBRARY ROUTINES CALLED....................NONE
OTHER LIBRARY ROUTINES CALLED....................NONE
CALLING ROUTINES.................................NONE
COMMONS REFERENCED..............................MSAREA

REMARKS:
THIS ROUTINE PROCESSES THE SUPERVISOR CALL TO FETCH
NODE COORDINATES. M IS AN ARBITRARY INDEX WITH WHICH
THE CALL IS MADE. N IS RETURNED AS THE NODE NUMBER OF
THE MTH NODE AND XX, YY ARE ITS COORDINATES. THE ROUT-
TINE IS CODED IN THIS FASHION TO MAKE IT SIMPLE TO
ALTER EXISTING PROGRAMS TO RUN UNDER THE FEHPOL SYSTEM
SINCE THEY USUALLY USE A RENUMBERING TABLE FOR BAND-
WIDTH OPTIMIZATION RATHER THAN ACTUALLY REORDERING
THE NODES.

INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300) ,Y(300)
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
N=NT(M)
XX=X(N)
YY=Y(N)
RETURN
END

**

SUBROUTINE FPNPMS(NUMEL,NUMCP,NUMNP,LTYPE,IBAND)

INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
N=NT(M)
XX=X(N)
YY=Y(N)
RETURN
END

REMARKS:
THIS ROUTINE PROCESSES THE SUPERVISOR CALL TO FETCH
MESH PARAMETERS. NUMEL=NUMBER OF ELEMENTS, NUMCP=
NUMBER OF CORNER POINTS, NUMNP=NUMBER OF POINTS IN-
CLUDING SIDE NODES, LTYPE=NUMBER OF NODES PER ELEMENT,
AND IBAND=SEMIBANDWIDTH.
INTEGER CY(6,395), NT(395), JM(395), MJ(3950), BW, TN
REAL X(300), Y(300)
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
NUMEL=NL
NUMCP=NC
NUMNP=NN
LTYPE=IE
IBAND=BW
RETURN
END

FUNCTION FPSPMS(STRING, LENGTH, NUM, IFLAG)
C
C SEGMENT NAME........................................FPSPMS
C SEGMENT TYPE.........................................FUNCTION
C SEGMENT LENGTH IN CARDS..............................34
C DATE OF LAST ALTERATION.............................20/07/76
C NUMBER OF LOCAL INTEGERS...........................5
C NUMBER OF LOCAL REALS..............................0
C
C FEHPOL ROUTINES CALLED................................NONE
C FORTRAN LIBRARY ROUTINES CALLED....................IFIX
C OTHER LIBRARY ROUTINES CALLED........................COMP
C CALLING ROUTINES.....................................FPPLST
C
C COMMONS REFERENCED..................................WKAREA, PARAMS
C
REMARKS:
C THIS ROUTINE PROCESSES THE SUPERVISOR CALL TO FETCH
C THE VALUE OF A 'SET' PARAMETER FROM THE PARAMETER POOL.
C IT IS CALLED AS FOLLOWS:
C -------CALL FPSPMS(13HPARAMETERNAME, 13, NUM, IFLAG)
C IF NUM>0 IT RETURNS THE VALUE OF THE PARAMETER ON ELEMENT
C OR NODE #NUM.  IF NUM<0 IT RETURN THE VALUE OF THE PARA-
C METER ON THE -NUMTH NODE OR ELEMENT ON WHICH IT HAS BEEN
C SET.  IF AN ERROR OCCURS A CODE IS RETURNED IN IFLAG AS
C FOLLOWS: IFLAG=1 THE PARAMETER DOES NOT EXIST, IFLAG=2
C NUM IS TOO NEGATIVE (NOT THAT MANY ASSIGNED VALUES),
C IFLAG=4 THE PARAMETER IS UNDEFINED ON THE NUMTH NODE OR
C ELEMENT.  OTHERWISE IFLAG=0 FETCH COMPLETE.
C
REAL A(5), STRING(10)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
LINDEX=0
DO 10 I=1, LS
INDEX=PS+(I-1)*NE+1
J=LENGTH
CALL COMP(J, PL(INDEX), 1, STRING(1), 1)
IF(J.NE.LENGTH)GO TO 10
CALL COMP(J, PL(INDEX), LENGTH+1, 2H, 1)
** SUBROUTINE FPSPUT(STRING, LENGTH, NUM, A) 

REAL STRING(10), A(1) 
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC 
REAL PL(1500), LN(20) 
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC 
COMMON/PARAMS/SI, DP, RP, LS, SV, DA, CR, PS, PT, CN, NE 
DO 10 I=1, NUM 

10 LINDEX = LINDEX + IFIX(PL(INDEX+NE-1)) 
   IFLAG = 1 
   GO TO 50 

20 IFLAG = 0 
   IF(NUM.LT.0) GO TO 30 
   FPSPMS = PL(PT+1-LINDEX-NUM) 
   IF(FPSPMS .EQ. -1111.0) IFLAG = 4 
   RETURN 

30 INDEX = IFIX(PL(INDEX+NE-1)) 
   J = 0 
   DO 40 I=1, INDEX 
      II = PT+1-LINDEX-1 
      IF(PL(II).NE. -1111.0) J = J+1 
      IF(J+NUM)40,50,40 
   40 CONTINUE 
   IFLAG = 2 
   GO TO 60 

50 FPSPMS = PL(II) 
50 RETURN 

END 

** SUBROUTINE FPSPUT(STRING, LENGTH, NUM, A) 

REAL STRING(10), A(1) 
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC 
REAL PL(1500), LN(20) 
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC 
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE 
DO 10 I=1, NUM
SUBROUTINE FPQUERY(STRING, LENGTH, NUMX, NUM)

C
C SEGMENT NAME......................FPQUERY
C SEGMENT TYPE......................SUBROUTINE
C SEGMENT LENGTH IN CARDS...........29
C DATE OF LAST ALTERATION...........20/07/76
C NUMBER OF LOCAL INTEGERS..........4
C NUMBER OF LOCAL REALS.............0
C FEHPOL ROUTINES CALLED.............NONE
C FORTRAN LIBRARY ROUTINES CALLED....IFIX
C OTHER LIBRARY ROUTINES CALLED.....COMP
C CALLING ROUTINES..................NONE
C COMMONS REFERENCED.................WKAREA
C Params
C
REMARKS:
C THIS ROUTINE PROCESSES THE SUPERVISOR CALL TO FIND
C OUT ABOUT 'SET' PARAMETERS. THE FIRST TWO ARGUMENTS
C ARE LIKE IN FPSPMS AND FPSPUT, IF NUMX>0 IT RETURNS
C IN NUM THE NODE OR ELEMENT NUMBER OF THE NUMXTH VALUE
C OF THE PARAMETER WHICH HAS BEEN SET. IF NUMX=0 OR NEG-
C RATIVE, THE NUMBER OF NODES OR ELEMENTS ON WHICH THE
C PARAMETER WAS SET IS RETURNED IN NUM. IF THE PARAMETER
C DOES NOT EXIST, NUM IS RETURNED=0.
C
REAL STRING(10)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
LINDEX=0
DO 10 I=1,LS
INDEX=PS+(I-1)*NE+1
J=LENGTH
CALL COMP(J,PL(INDEX),1,STRING(1),1)
IF(J.NE.LENGTH)GO TO 10
CALL COMP(J,PL(INDEX),LENGTH+1,2H,1)
IF(J.GE.1)GO TO 20
10 LINDEX=LINDEX+IFIX(PL(INDEX+NE-1))
NUM=0
RETURN
SUBROUTINE FPSSTM
SEGMEN NAME .......... FPSSTM
SEGMEN TYPE .......... SUBROUTINE
SEGMEN LENGTH IN CARDS ........ 49
DATE OF LAST ALTERATION ...... 20/07/76
NUMBER OF LOCAL INTEGERS .......... 1
NUMBER OF LOCAL REALS .......... 0
FEHPOL ROUTINES CALLED .......... FPWARN
FPNWCD
FPCMND
FORTRAN LIBRARY ROUTINES CALLED ........ NONE
OTHER LIBRARY ROUTINES CALLED .......... COMP
COPY...
CALLING ROUTINES .......... FPSTOP
FPQUST
COMONS REFERENCED .......... WKAREA
PARAMS

REMARKS:
THIS ROUTINE IS THE MAIN ENTRY TO THE SYSTEM. AT
INITIAL PROGRAM LOAD TIME IT INITIALIZES THE POOL
SIZES, DEVICE TABLE, CHARACTER POINTER, SKIP AND
TRACE FLAGS. IN ADDITION, IT CALCULATES THE IM-
PORTANT LOCAL INSTALLATION PARAMETERS SUCH AS CPR,
THE NUMBER OF CHARACTERS WHICH FIT IN A FORTRAN REAL
VARIABLE ON THE LOCAL MACHINE; NE, THE NUMBER OF REAL
FORTRAN VARIABLES REQUIRED FOR SYMBOL AND PARAMETER
POOL ENTRIES TO ALLOW NAMES OF LENGTH CN. THE POOL
SYMBOL INDICES AND DATA POINTERS ARE INITIALIZED.
AFTER INITIALIZATION IT LOOP ON SEARCH FOR A COMMAND,
WRITING THE SYNTAX ERROR MESSAGE IF EVER AN ERROR EXIT
IS TAKEN FROM FPCMND.
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IF(PT.EQ.1500)GO TO 40

INITIAL FEHPOL DEVICE ASSIGNMENTS

DP = DISPLAY RP = REPLY SI = SOURCE LO = LISTING DA = DIRECT
DP = 6, RP = 6
SI = 5
LO = 7
DA = 4

INITIALIZE FLAGS
SK = 1
TC = 0

INITIALIZE POOL PARAMETERS
PS = SYMBOL POOL SIZE
PT = PS + PARAMETER POOL SIZE
KS = CURRENT NUMBER OF SYMBOLS
KP = ENTRY POINTER
LS = CURRENT NUMBER OF PARAMETERS
LP = ENTRY POINTER
CR = NUMBER OF CHARACTERS PER REAL VARIABLE
CN = MAXIMUM NUMBER OF CHARACTERS RECOGNIZED IN A NAME
NE = NUMBER OF REAL VARIABLES REQUIRED FOR A POOL ENTRY
PS = 500
DO 70 I = 1, PS
70 PL(I) = -1.0
PT = 1500
C * CALCULATE CR FOR THIS MACHINE
CALL COPY(8, LN(1), 1, 8H........, 1)
CALL COPY(1, LN(2), 1, 2H**, 1)
CR = 0
50 CR = CR + 1
1 = 1
CALL COMP(I, LN(1), CR + 1, 2H**, 1)
IF(I.NE.1)GO TO 50
IF(CR.NE.4.AND.CR.NE.6.AND.CR.NE.8)STOP 01
C *
CN = 16
NE = (CN + CR - 1) / CR + 1
KP = PS - 1
KS = 0
LS = 0
LP = PT
C

INITIALIZE CHARACTER POINTER
IP = 1
C
WRITE(LO, 24)
BEGIN COMMAND INTERPRETATION

40 CALL FPNWCD
10 IF(FPCMND(0))20,10,20
20 CALL FPWARN(1)
GO TO 40

24 FORMAT(1H1///)
END

**

FUNCTION FPCMND(N)

Segue NAME.................................FPCMND
Segue TYPE.................................FUNCTION
Segue LENGTH IN CARDS....................24
Segue DATE OF LAST ALTERATION...........20/07/76
Segue NUMBER OF LOCAL INTEGERS.........0
Segue NUMBER OF LOCAL REALS............0

FEHPOL ROUTINES CALLED..................FPASST
                                    FPCUST
                                    FPDLST
                                    FPEXST
                                    FPINST
                                    FPKPST
                                    PPLEST
                                    FPMEST
                                    FPNO

FORTRAN LIBRARY ROUTINES CALLED........NONE
OTHER LIBRARY ROUTINES CALLED...........NONE
CALLING ROUTINES.........................FPSSTM
COMMONS REFERENCED......................NONE

REMARKS:
THIS ROUTINE SEARCHES FOR EACH OF THE POSSIBLE
COMMAND TYPES IN AN ORDER SPECIFIED BY THE COMMAND
FREQUENCY TABLE (BELOW), SO THAT THE BEST RESPONSE
IS AVAILABLE ON THE MOST USED COMMAND TYPES. IT
REPRESENTS THE FOLLOWING BNF RULES:
<COMMAND>::=<ASSIGNMENT STATEMENT>
<COMMAND>::=<CURVE STATEMENT>
<COMMAND>::=<DELETE STATEMENT>
<COMMAND>::=<EXECUTE STATEMENT>
<COMMAND>::=<GENERATE STATEMENT>
<COMMAND>::=<INTERPOLATE STATEMENT>
<COMMAND>::=<KEEP STATEMENT>
<COMMAND>::=<LET STATEMENT>
<COMMAND>::=<MESH STATEMENT>
<COMMAND>::=<NODES STATEMENT>
<COMMAND>::=<PLOT STATEMENT>
<COMMAND>::=<REGION STATEMENT>
<COMMAND>::=<SET STATEMENT>
<COMMAND>::=<TRACE STATEMENT>
<COMMAND>::=<UNKEEP STATEMENT>
<COMMAND>::=<WRITE STATEMENT>

LOOK FOR A COMMAND

A....ASSIGN  H....  O....  V....
B....  I....INTERPOLATE  P....PLOT  W....WRITE
C....CURVE  J....  Q....QUIT  X....
D....DELETE  K....KEEP  R....REGION  Y....
E....EXECUTE  L....LET  S....SET  Z....
F....
G....GENERATE  N....NODS  U....UNKEEP

GO TO 3
1 IF(FPASST(0))700,800,11
3 IF(FPCUST(0))700,800,9
4 IF(FPDLST(0))700,800,23
5 IF(FPEXST(0))700,800,16
7 IF(FPGEST(0))700,800,13
9 IF(FPINST(1))700,800,18
11 IF(FPKPST(0))700,800,21
12 IF(FPLEST(0))700,800,17
13 IF(FPMEST(0))700,800,5
14 IF(FPNOST(0))700,800,19
16 IF(FPPLST(0))700,800,1
17 IF(FPQUST(0))700,800,900
18 IF(FPREGST(0))700,800,14
19 IF(FPSEST(0))700,800,4
20 IF(FPTCST(0))700,800,12
21 IF(FPUKST(0))700,800,20
23 IF(FPWRST(0))700,800,7
700 FPCMND=-1.0
RETURN
800 FPCMND=0.0
RETURN
900 FPCMND=1.0
RETURN
END

** SUBROUTINE FPACC(S(INTABLE,NUMLIST,INDEX,NMENTS) **

SEGMENT NAME.............................FPACC
SEGMENT TYPE............................SUBROUTINE
SEGMENT LENGTH IN CARDS..................20
DATE OF LAST ALTERATION..............20/07/76
NUMBER OF LOCAL INTEGERS.............1
NUMBER OF LOCAL REALS................0
FEHPOL ROUTINES CALLED...............NONE
FORTRAN LIBRARY ROUTINES CALLED............IFIX
OTHER LIBRARY ROUTINES CALLED..........NONE
CALLING ROUTINES

FPDLST
FPMEST
FPWRST
FPPLST
FPSEST
FPS3SC
FPINTP

COMMONS REFERENCED

WKAREA
PARAMS

REMARKS:

THIS ROUTINE IS USED TO ACCESS EITHER OF THE POOLS.
IT IS USED IN CONJUNCTION WITH FPDTNM IN THAT THE
TEMPORARY VARIABLE TM IN COMMON WKAREA IS ALWAYS
SET WHENEVER A DEFINE TYPE NAME IS SCANNED TO THE
SYMBOL INDEX (1ST SYMBOL=0, 2ND=1, ETC). A SUBSEQUENT
CALL TO FPACCS WITH ONLY THE POOL NUMBER (1=SUBLST,
2=PARAMETERS) RETURNS NUMLST=THE SYMBOL INDEX, INDEX=
THE DATA INDEX WHERE IT STARTS + 1 (NOTE: THE DATA
GOES BACKWARD FROM THE END OF THE POOLS) AND NMENTS=
THE NUMBER OF VALUES OR POINTS THAT THE SYMBOLS REP-
RESENTS. CAUTION SHOULD BE USED IN MAKING CALLS TO
THIS ROUTINE AS SOMETIMES TM IS RESET FROM A SAVED
VALUE FROM AN EARLIER SCANNED FPDTNM (IE. IN FPPLST
TO PLOT THE INTERNAL BOUNDARIES THE TM'S ARE SAVED
IN ARRAY SV).

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
NUMLST=IFIX(TM)
INDEX=0
NMENTS=0
GO TO (10, 60), ITABLE
10 IF(NUMLST)50, 40, 20
20 DO 30 1=1, NUMLST
30 INDEX=INDEX+IFIX(PL(1*NE))
40 NMENTS=IFIX(PL(NUMLST*NE+NE))
50 RETURN
60 IF(NUMLST)50, 90, 70
70 DO 30 1=1, NUMLST
80 INDEX=INDEX+IFIX(PL(PS+1*NE))
90 NMENTS=IFIX(PL(PS+NUMLST*NE+NE))
RRETURN
END

**

FUNCTION FPINRG(XX, YY, NMPTS, INDEX)

SEGMENT NAME

FPINRG

SEGMENT TYPE

FUNCTION

SEGMENT LENGTH IN CARDS

35

DATE OF LAST ALTERATION

20/07/76

NUMBER OF LOCAL INTEGERS

4

NUMBER OF LOCAL REALS

7
FEHPOL Routines Called: None

Fortran Library Routines Called: None

Other Library Routines Called: None

Calling Routines: FPGEST, FPSEST

Commons Referenced: WKAREA, PARAMS, MSAREA

Remarks:
This routine used the FEHPOL data structure to test if a point (XX, YY) in the simply connected region whose points are located in reverse order in the symbol pool starting from index-1. (i.e., X(1) = PL(INDEX-1), Y(1) = PL(INDEX-2), X(2) = PL(INDEX-3)...) where X(I), Y(I) are the coordinates of the points defining the region boundary.

```
INTEGER CY(6,395), NT(395), JM(395), MJ(3950), BW, TN
REAL X(300), Y(300)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
COMMON/MSAREA/CY, NT, JM, X, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)

NODE(IX, I) = PS - 2 * (IX + I)
NUMX = 0
JJ = NODE(INDEX, NMPTS)
DO 10 I = 1, NMPTS
   II = NODE(INDEX, I)
   XI = PL(JJ)
   Y1 = PL(JJ-1)
   X2 = PL(II)
   Y2 = PL(II-1)
   DELTX = X2 - XI
   IF(DELTX)60, 10, 60
   60 S = (Y2 - Y1) / DELTX
   IF((YY - Y1) - S * (XX - XI))30, 40, 10
   40 IF(XX .GE. XI .AND. XX .LE. X2) GO TO 50
   IF(XX .LE. X1 .AND. XX .GE. X2) GO TO 50
   GO TO 10
   30 IF(XX .LE. X1 .AND. XX .LE. X2) GO TO 10
   IF(XX .GT. X1 .AND. XX .GT. X2) GO TO 10
   NUMX = NUMX + 1
   10 JJ = II
   NUMX = NUMX - 2 * (NUMX / 2)
   IF(NUMX .EQ. 1) GO TO 20
   20 FPINRG = 1.0
   RETURN
   50 FPINRG = 0.0
   RETURN
END
```

Subroutine FPENTR(ITABLE, JP)
**FUNCTION FPDLST(N)**

** INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
 REAL PL(1500),LN(20)
 COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
 COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
 LENGTH=(NE-1)*CR
 GO TO (20,10),ITABLE
 10 I=PS+LS*NE+1
 CALL COPY(LENGTH,PL(I),1,1,32H)
 LENGTH=IP-JP
 CALL COPY(LENGTH,PL(I),1,1,1,JP)
 LS=LS+1
 RETURN
 20 I=KS*NE+1
 CALL FPDLET(KS)
 LENGTH=IP-JP
 IF(LENGTH.GT.CN-1)LENGTH=CN-1
 CALL COPY(LENGTH,PL(I),2,1,JP)
 KS=KS+1
 RETURN
END
FEHPOL ROUTINES CALLED........................FPTRAC
                        FPAACS
                        NEXTIS
                        FPDLET

FORTRAN LIBRARY ROUTINES CALLED................NONE

OTHER LIBRARY ROUTINES CALLED................NONE

CALLING ROUTINES.................................FPCMND

COMMONS REFERENCED..............................WKAREA
                        PARAMS

REMARKS:
THIS STATEMENT REPRESENTS THE FOLLOWING BNF RULES:
<DELETE STATEMENT>::= DELETE <DEFINED TYPE NAME>

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
CALL FPTRAC(6HFPDLST)
 IF(NEXTIS(6HDELETE,6))700,10,900.
   10 IF(FPDTNM(0))700,20,700
   20 CALL FPACCS(1,NUMLST,INDEX,I)
   IF(NUMLST.GE.0)CALL FPDLET(NUMLST)
   GO TO 800

700 FPDLST=-1.0
RETURN
800 FPDLST=0.0
RETURN
900 FPDLST=1.0
RETURN
END

**
FUNCTION FPTCST(I)

SEGMENT NAME.........................FPTCST
SEGMENT TYPE.........................FUNCTION
SEGMENT LENGTH IN CARDS.............22
DATE OF LAST ALTERATION............20/07/76
NUMBER OF LOCAL INTEGERS............0
NUMBER OF LOCAL REALS..............0

FEHPOL ROUTINES CALLED........................FPTRAC
                        FPSKIP
                        NEXTIS

FORTRAN LIBRARY ROUTINES CALLED................NONE

OTHER LIBRARY ROUTINES CALLED................NONE

CALLING ROUTINES.................................FPCMND

COMMONS REFERENCED..............................WKAREA
                        PARAMS

REMARKS:
THIS STATEMENT REPRESENTS THE BNF RULES:

<TRACE STATEMENT>::= TRACE ON

<TRACE STATEMENT>::= TRACE OFF

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HPTCST)
IF(NEXTIS(6HTRACE, 5))760, 10, 900
10 IF(NEXTIS(2H0N, 2, ))700, 40, 20
20 IF(NEXTIS(4H0FF, 3, ))700, 50, 700
40 CALL FPSKIP
   TC=1
   GO TO 800
50 CALL FPSKIP
   TC = 0
   GO TO 800
700 FPTCST=-1.0
   RETURN
800 FPTCST=0.0
   RETURN
900 FPTCST=1.0
   RETURN
END

FUNCTION FPINST(N)
SEGMENT NAME..............................FPINST
SEGMENT TYPE............................FUNCTION
SEGMENT LENGTH IN CARDS...............35
DATE OF LAST ALTERATION..............20/07/76
NUMBER OF LOCAL INTEGERS..............2
NUMBER OF LOCAL REALS................1
FEHPOL ROUTINES CALLED.................FPTRAC
                                      FPDTNM
                                      NEXTIS
                                      FPSKIP
                                      FPRLNM
                                      FPINTP
FORTRAN LIBRARY ROUTINES CALLED........IFI
OTHER LIBRARY ROUTINES CALLED.........NONE
CALLING ROUTINES.......................FPCMND
COMMONS REFERENCED...................WKAREA
                                      PARAMS
REMARKS:
THIS ROUTINE REPRESENTS THE BNF RULES:

<INTERPOLATE STATEMENT>::= INTERPOLATE <DEFINED TYPE NAME> N
  INTERPOLATE <DEFINED TYPE NAME> NODES=<REAL NUMB
<INTERPOLATE STATEMENT>::= INTERPOLATE <DEFINED TYPE NAME> NODES=<REAL NUM
<INTERPOLATE STATEMENT>::=
INTERPOLATE <DEFINED TYPE NAME> NODES=<REAL NUMB>

<INTERPOLATE STATEMENT>::=

INTERPOLATE <DEFINED TYPE NAME> NODES=<REAL NUMB>

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
CALL FPTRAC(6HFPINST)
IF(NEXTIS(12HINTERPOLATE,11))700,10,900
10 IF(FPDTNM(0))700,20,700
20 I=NEXTIS(6HNODES,5)
I=NEXTIS(2H=,1)
CALL FPSKIP
A=TM
IF(FPRLNM(0))700,30,700
30 I=IFIX(TM)
TM=A
JP=NEXTIS(2H,1)
IF(NEXTIS(6HLINEAR,6))700,50,40
40 IF(NEXTIS(8HCIRCULAR,8))700,60,55
55 IF(NEXTIS(6HSMOOTH,6))700,56,70
70 IF(NEXTIS(8HREGULAR,7))700,57,70
56 CALL FPINTP(I,3)
GO TO 800
50 CALL FPINTP(I,1)
GO TO 800
60 CALL FPINTP(I,2)
GO TO 800
57 CALL FPINTP(I,4)
GO TO 800
700 FPINST=-1.0
RETURN
800 FPINST=0.0
RETURN
900 FPINST=1.0
RETURN
END

**

SUBROUTINE FPDLET(NUMLST)

SEGMENT NAME..............................FPDLET
SEGMENT TYPE..............................SUBROUTINE
SEGMENT LENGTH IN CARDS.......................13
DATE OF LAST ALTERATION......................20/07/76
NUMBER OF LOCAL INTEGERS....................0
NUMBER OF LOCAL REALS.......................0
FEHPOL ROUTINES CALLED......................NONE
FORTRAN LIBRARY ROUTINES CALLED...............IFIX
OTHER LIBRARY ROUTINES CALLED...............COPY
CALLING ROUTINES..........................FPDLST
FPENTR
COMMONS REFERENCED........................WKAREA
PARAMS

REMARKS:
THIS ROUTINE RENAMES THE 'NUMLST' TH SYMBOL TO NULL OR IF IT IS THE LAST SYMBOL IN THE POOL, ACTUALLY DECREASES THE POINTERS AND DELETES IT.

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IF(NUMLST.EQ.KS-1) GO TO 10
CALL COPY((NE-1)*CR-1, PL(NUMLST*NE+1), 2, 1)
RETURN
10 KP=KP+IFIX(PL(KS*NE))
KS=KS-1
RETURN
END

FUNCTION FPDFIN(N)

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPDFIN)
KK=KP
IF(FPLMNT(0))700,25,700
25 IF(FPLMNT(0))700,25,30
30 IF(FPIDFR(0))700,40,700
40 IF(KK-KP.LE.2) GO TO 400
I=KK

REMARKS:
THIS STATEMENT REPRESENTS THE FOLLOWING BNF RULES:
<DEFINITION> ::= <LIST OF ELEMENTS> <IDENTIFIER>

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPDFIN)
KK=KP
IF(FPLMNT(0))700,25,700
25 IF(FPLMNT(0))700,25,30
30 IF(FPIDFR(0))700,40,700
40 IF(KK-KP.LE.2) GO TO 400
I=KK
J=KK
100 I=I-2
200 J=J-2
   IF(J.LT.KP)GO TO 300
   PL(I)=PL(J)
   PL(I+1)=PL(J+1)
   IF(PL(J).NE.PL(J-2))GO TO 100
   IF(PL(J+1).NE.PL(J-1))GO TO 100
   GO TO 200
300 KP=I+2
   IF(PL(KP).EQ.PL(KK-2).AND.PL(KP+1).EQ.PL(KK-1))
   1 KP=KP+2
   IF(KK.EQ.KP)KP=KK-2
400 PL(KS*NE)=FLOAT((KK-KP)/2)
   GO TO 800
700 FPDFIN=-1.0
   KP=KK
   RETURN
800 FPDFIN=0.0
   RETURN
900 FPDFIN=1.0
   .RETURN
   END

**
FUNCTION FPMEST(N)
C
C SEGMENT NAME.................................FPMEST
C SEGMENT TYPE.................................FUNCTION
C SEGMENT LENGTH IN CARDS.......................93
C DATE OF LAST ALTERATION.....................25/07/76
C NUMBER OF LOCAL INTEGERS....................13
C NUMBER OF LOCAL REALS......................10
C
C PEHPOL ROUTINES CALLED........................FPTRAC
C NEXTIS
C FPDTNM
C FPACCS
C FPWARN
C FPSTOP
C FPCNCT
C FPNWCD
C
C FORTRAN LIBRARY ROUTINES CALLED...............IFIX
C  .FLOAT
C OTHER LIBRARY ROUTINES CALLED................COMP
C
C CALLING ROUTINES............................FPCMND
C
C COMMONS REFERENCED..........................WKAREA
C  .PARAMS
C  .MSAREA
C
C REMARKS:
C THIS STATEMENT REPRESENTS THE BNF RULES:
C <MESH STATEMENT>::=
C  .  MESH <DEFINED TYPE NAME> WITH THREE NODED TRIANGLES
C <MESH STATEMENT>::=
C  .  MESH <DEFINED TYPE NAME> WITH SIX NODED TRIANGLES
C \text{MESH STATEMENT}::= \\
C \text{MESH} \text{<DEFINED TYPE NAME>} \text{HOLES=\text{(<DEFINED TYPE NAME>),...}} \\
C \text{WITH THREE NODED TRIANGLES} \\
C \text{MESH STATEMENT}::= \\
C \text{MESH} \text{<DEFINED TYPE NAME>} \text{HOLES=\text{(<DEFINED TYPE NAME>),...}} \\
C \text{WITH SIX NODED TRIANGLES} \\
\text{IT PICKS OUT THE REGIONS FROM THE POOL AND SETS UP THE} \\
\text{MESH AREA (X,Y) ARRAY STARTING WITH EXTERNAL BOUNDARY,} \\
\text{1ST INTERNAL BOUNDARY (IF ANY), 2ND, 3RD, ... , THE INTERNAL} \\
\text{NODES. THEN IT CALLS THE CONNECTION ROUTINE.} \\
C 
\begin{verbatim}
INTEGER Cy(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/3I,DP,RP,L0,SV,DA,CR,PS,PT,CN,NE
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
NODE(INDEX,I)=PS-2*(INDEX+I)
CALL FPTRAC(6HFPMEST)
\end{verbatim}

\text{IF(NEXTIS(4HMESS,4))700,10,90} \\
10 \text{IF(FPDTNM(0))700,20,700} \\
20 \text{CALL FPACCS(1,IT,NI,NP(1))}
SV(1)=IT
\text{IF(NP(1).LT.3)CALL FPWARN(9)}
\text{IF(IT.LT.0.OR.NP(1).LT.3)GO TO 200}
TN=0
DO 24 I=1,NP(1)
II=NODE(NI,I)
TN=TN+1
X(TN)=PL(II)
Y(TN)=PL(II-1)
24 \text{IF(NEXTIS(2H,1))700,100,140} \\
100 \text{IF(FPDTNM(0))700,120,700} \\
120 M=M+1
\text{CALL FPACCS(1,JT,NI,NP(M))}
SV(M)=JT
\text{IF(NP(M).LT.3)CALL FPWARN(14)}
\text{IF(JT.LT.0.OR.NP(M).LT.3)GO TO 200}
DO 150 I=1,NP(M)
II=NODE(NI,I)
TN=TN+1
X(TN)=PL(II)
Y(TN)=PL(II-1)
150 \text{IF(NEXTIS(2H,1))700,100,130} \\
130 \text{IF(NEXTIS(2H,1))700,140,700} \\
140 \text{NB=M}
26 \text{IF(NEXTIS(4HWITH,4))700,30,25} \\
25 \text{IF(NEXTIS(4HTHREE,5))700,40,700} \\
30 \text{I=3}
\text{IE=3}
\text{GO TO 50}
\end{verbatim}
40 I=6
IE=6
50 IF(NEXTIS(6HNODED,5))700,60,700
60 IF(NEXTIS(10HTRIANGLES,9))700,70,700
70 NIN=0
NLIST=0
INDF=3951
DO 71 I=1,KS
M=1
K=(I-1)*NE+1
CALL COMP(M,PL(K),1,2H:1)
IF(M.NE.1)GO TO 74
M=(NE-1)*CR-1
II=M
CALL COMP(M,PL(K),2,PL(IT*NE+1),2)
IF(M.NE.II)GO TO 74
K=K+NE-1
P=PL(K)
IF(P.LE.0.0)CALL FPSTOP(31)
NODES=IFIX(P)
IFLGSM=1
IF(P-FLOAT(NODES).EQ.0.5)IFLGSM=-1
DO 72 J=1,NODES
INDF=INDF-1
MJ(INDF)=IFLGSM
II=NODE(NLIST,J)
NIN=NIN+1
X(TN+NIN)=PL(II)
Y(TN+NIN)=PL(II-1)
72 CONTINUE
74 P=PL(NE*I)
IF(P.LE.0.0)CALL FPSTOP(31)
71 NLIST=NLIST+IFIX(P)
90 NC=TN+NIN
C *********************************************
C * FIND A POINT BEHIND STARTING SEGMENT FOR *
C * FPMESH. *
C *********************************************
XI=X(1)
YI=Y(1)
XJ=X(NP(1))
YJ=Y(NP(1))
DD=SQR((XJ-XI)**2+(YJ-YI)**2)/1000.0
XC=(XJ+XI)/2.0
YC=(YJ+YI)/2.0
A=XJ-XI
IF(A)250,211,250
250 Y1=YC+DD
X1=XC
Y2=YC-DD
X2=XC
GO TO 220
210 Y1=YC
X1=XC+DD
Y2=YC
X2=XC-DD
220 IF(FPINRG(X1,Y1,NP(1),INDX1))240,230,240
FUNCTION FPWRST(N)

INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
INTEGER DV
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
CALL FPTRAC6HFPWRST)
CALL FPTable
IF(NEXTIS(6HWRITE  ,5))700,10,900
10 IF(NEXTIS(10HPARAMETERS,10))700,50,20
20 IF(NEXTIS(12HCONNECTIVITY,12))700,60,30
30 IF(NEXTIS(8HSYMBOLS ,7))700,70,35
35 IF(NEXTIS(10HMESHESPECs ,9))700,77,37
37 IF(NEXTIS(6HNODAL  ,5))700,38,40
38 IF(NEXTIS(12HCOORDINATES  ,11))700,39,900
40 JP=IP
IF(FPDTNM1))700,41,900
41 CALL PPACCs(2,IWF,I,K)
IWF=IWF+1
IF(IWF)700,42,51
42 IF=JP
IF(FPDTNM(0))700,80,900
50 IWF=-1
51 K=0
IF(LS.EQ.0)GO TO 200
WRITE(DP,1005)
DO 23 I=1,LS
II=PS+(I-1)*NE
JJ=IFIX(PL(II+NE))
IF(IWF.NE.1.AND.IWF.NB.1)GO TO 23
IF(CR.EQ.4)WRITE(DP,1003)(PL(II+J),J=1,NE-1)
IF(CR.EQ.6)WRITE(DP,1019)(PL(11+J),J=1,NE-1)
IF(CR.EQ,8)WRITE(DP,1020)(PL(II+J),J=1,NE-1)
IZ = 0
DO 300 J=1,JJ
IF(PL(PT+1-K-J).EQ.-1111.0)00  TO  300
IZ=IZ+1
MJ(IZ)=J
300 CONTINUE
WRITE(DP,1006)(MJ(J),PL(PT+1-K-MJ(J)),J=1,IZ)
WRITE(DP,1011)
23 K=K+JJ
200 K=LP-(PS+NE*LS)
WRITE(DP,1002).L,S,K
GO TO 100
60 IF(NL)63,63,61
61 WRITE(DP,1008)
IF(IE.EQ.3)WRITE(DP,1009)(I,(CY(J,I),J=1,3),I=1,NL)
IF(IE.EQ.6)WRITE(DP,1010)(I,(CY(J,I),J=1,6),I=1,NL)
39 WRITE(DP,1011)
WRITE(DP,1012)
WRITE(DP,1001)(I,X(I),Y(I),I=1,NN)
77 WRITE(DP,1004)NL,NC,NN,BW
GO TO 100
63 CALL FPWARM(11)
GO TO 100
70 IF(KS)74,74,71
71 K=2
M=0
WRITE(DP,1014)
IBEGIN=1
IEND=KS
72 DO 75 I=IBEGIN,IEND
JJ=(I-1)*NE
IF(CR.EQ.4) WRITE(DP,1016)(PL(JJ+J),J=1,NE-1)
IF(CR.EQ.6) WRITE(DP,1019)(PL(JJ+J),J=1,NE-1)
IF(CR.EQ.8) WRITE(DP,1020)(PL(JJ+J),J=1,NE-1)
P=PL(I*NE)
M=IFIX(P)*2+M
II=0
DO 73 K=K,M,2
II=II+1
73 WRITE(DP,1015)II,PL(PS-K),PL(PS-K-1)
K=M+2
75 WRITE(DP,1011)
K=KP-KS*NE
WRITE(DP,1013)KS,K
GO TO 100
74 CALL FPWARN(12)
GO TO 100
1001 FORMAT(IH,I4,4X,2F10.4,6X,I4,4X,2F10.4)
1002 FORMAT(10X,21HNUMBER OF PARAMETERS=,I6/
110X,21HNUMBER OF ELEMENTS= ,I6/
120X,16HNUMBER OF NODES=,5X,I6/
130X,10HBANDWIDTH=,11X,I6/)  
1005 FORMAT(18H PARAMETER TABLE:,2X,3(3HN/E,3X,5HVALUE,6X)/)
1006 FORMAT(18X,I4,E13.6,I4,E13.6,I4,E13.6)
1008 FORMAT(21H CONNECTIVITY TABLE:/)
1009 FORMAT(1H,I4,4X,3I4,6X,I4,4X,3I4,6X,I4,4X,3I4)
1010 FORMAT(1H,I4,4X,6I4,6X,I4,4X,6I4)
1012 FORMAT(41H NODAL COORDINATES X-COORD Y-COORD /)
1013 FORMAT(10X,21HNUMBER OF SYMBOLS= ,I6/
110X,21HSYMBOL SPACE LEFT= ,I6/)  
1014 FORMAT(15H SYMBOL TABLE:,11X,7HX-COORD,9X,7HY-COORD/)  
1015 FORMAT(1H,12X,I4,4X,1PE16.9,4X,1PE16.9)
1016 FORMAT(1H,4A4)
1017 FORMAT(/)
1018 FORMAT(1H,24X,7HX-COORD,9X,7HY-COORD/)  
1019 FORMAT(1H,9A6)
1020 FORMAT(1H,8A8)
80 CALL FPACCS(1,IBEGIN,NI,M)
IF( IBEGIN.LT.0)GO TO 100
M=2*NI
K=M+2
IBEGIN=IBEGIN+1
IEND=IBEGIN
WRITE(DP,1018)
GO TO 72
100 IF(NEXTIS(2H,1))700,10,800
700 FPWRST=-1.0
RETURN
800 FPWRST=0.0
RETURN
**

FUNCTION FPPLST(N)

FUNCTION FPPLST

SEGMENT NAME.............................................. FPPLST

SEGMENT TYPE.............................................. FUNCTION

SEGMENT LENGTH IN CARDS................................ 160

DATE OF LAST ALTERATION............................... 20/07/76

NUMBER OF LOCAL INTEGERS.............................. 22

NUMBER OF LOCAL REALS................................. 29

FEHPOL ROUTINES CALLED................................. FPTRAC

FPSPMS

NEXTIS

FPDTNM

FPACCS

FPSTOP

FPRLMN

FPWARN

FPCNPL

FORTRAN LIBRARY ROUTINES CALLED...................... ABS

.FLOAT

ADVANCE

.COMP

VECTOR

SQUARE

BOXNUM

OTHER LIBRARY ROUTINES CALLED...................... FPCMND

CALLING ROUTINES....................................... WKAREA

PARAMS

MSAREA

PLAREA

COMMONS REFERENCED.................................... WKAREA

PARAMS

MSAREA

PLAREA

REMARKS:

THIS ROUTINE REPRESENTS THE FOLLOWING BNF RULES:

<PLOT STATEMENT>::=PLOT MESH

<PLOT STATEMENT>::=PLOT AND LABEL MESH

<PLOT STATEMENT>::=PLOT NODES

<PLOT STATEMENT>::=PLOT CONTOUR MAP OF <DEFINED TYPE NAME>,

. LEVELS=(<REAL NUMBER>,....)

<PLOT STATEMENT>::=PLOT <DEFINED TYPE NAME>

INTEGER NW(6)

INTEGER CY(6,395), NT(395), JM(395), MJ(3950), BW, TN

REAL CONLEV(15)

REAL X(300), Y(300)

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC

REAL PL(1500), LN(20)

COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC

COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE

COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)

COMMON/PLAREA/BD, SC, PW

DATA NW(1), NW(2), NW(3), NW(4), NW(5), NW(6)/4, 5, 6, 2, 3, 1/

NODE(INDEX, I)=PS-2*(INDEX+I)
CALL FPTRAC(6HFPPPLST)
IW=0
BD=23.0
PW=750.0
IF(NEXTIS(4HPLOT,4))700,35,900
35 I=0
XMAX=FPSPMS(4HMAX,4,1,I)
IF(I.NE.0)XMAX=PW
I=0
YMAX=FPSPMS(4HMAX,4,1,I)
IF(I.NE.0)YMAX=PW
I=0
XMIN=FPSPMS(4HMIN,4,1,I)
IF(I.NE.0)XMIN=0.0
I=0
YMIN=FPSPMS(4HMIN,4,1,I)
IF(I.NE.0)YMIN=0.0
SC=PW/(XMAX-XMIN)
CALL ADVANCE((YMAX-YMIN)*SC+BD)
LABEL=1
LZ=NEXTIS(4HAND ,3)
IF(NEXTIS(6HLABEL , 5 ) ) 700 , 400 , 410
400 LABEL=2
410 CONTINUE
IF(NEXTIS(4HMESH,4))700,10,36
36 IF(NEXTIS(6HNODES ,5))700,51,140
140 IF(NEXTIS(8HCONTOURS,8))700,141,149
141 I=NEXTIS(4HMAP ,3)
I=NEXTIS(2HOF,2)
IF(FPDTNM(1))700,142,700
142 CALL FPACCs(2,LIST,INDEX,NUM)
IF(LIST.LT.0)GO TO 800
IF(NUM.NE.NN)CALL FPSTOP(54)
I=NEXTIS(2H, ,1)
I=NEXTIS(6HLEVELS,6)
I=NEXTIS(2H= ,1)
I=NEXTIS(2H( ,1)
II=0
143 IF(FPRLNM(0))700,144,146
144 II=II+1
IF(II.GT.15)GO TO 146
CONLEV(II)=TM
I=NEXTIS(2H, ,1)
GO TO 143
146 I=NEXTIS(2H),1)
IF(NL.LT.1)CALL FPWARN(11)
IF(NL.LT.1)GO TO 800
CALL FPCNPL(CONLEV,II,INDEX-1)
IW=1
1010 TM=FLOAT(SV(IW))
GO TO 150
149 IF(FPDTNM(0))700,150,700
150 CALL FPACCs(1,LIST,INDEX,NUM)
IF(LIST.LT.0)GO TO 800
Q=PL(LIST*NE+1)
LZ=1
CALL COMP(LZ,Q,1,2H@ ,1)
IF(LZ.EQ.1) GO TO 170
LZ=1
CALL COMP(LZ,Q,1,2H,1)
GO TO 180

170 II=NODE(INDEX,1)
JJ=NODE(INDEX,NUM)
X1=PL(II)\*SC+BD-XMIN
X2=PL(JJ)\*SC+BD-XMIN
Y1=PL(II-1)\*SC+BD-YMIN
Y2=PL(JJ-1)\*SC+BD-YMIN
CALL VECTOR(X1,Y1,X2,Y2)
LZ=0

180 DO 160 I=1,NUM-1
II=NODE(INDEX,I)
X1=PL(II)\*SC+BD-XMIN
X2=PL(II-2)\*SC+BD-XMIN
Y1=PL(II-1)\*SC+BD-YMIN
Y2=PL(II-3)\*SC+BD-YMIN
CALL SQUARE(X1,Y1)
IF(LZ.EQ.0) CALL VECTOR(X1,Y1,X2,Y2)
IF(NUM.GT.1) CALL SQUARE(X2,Y2)
160 IF(IW.EQ.0) CALL VECTOR(X1,Y1,X2,Y2)
1000 IW=IW+1
10 IFN=NT(1)
ILN=NT(TN)
NMSEGS=0
MJ(1)=0
JM(1)=0
LL=6-IE
DO 50 I=1,NL
J=1
DO 40 L=1,IE
K=NW(LL+J)
JJ=CY(J,I)
KK=CY(K,I)
DO 90 I=1,NMSEGS
IF(JJ.EQ.MJ(II).AND.KK.EQ.JM(II)) GO TO 40
90 CONTINUE
NMSEGS=NMSEGS+1
MJ(NMSEGS)=JJ
JM(NMSEGS)=KK
30 X1=X(JJ)\*SC+BD-XMIN
X2=X(KK)\*SC+BD-XMIN
Y1=Y(JJ)\*SC+BD-YMIN
Y2=Y(KK)\*SC+BD-YMIN
GO TO (420,430), LABEL
420 CONTINUE
DX=0.0
DY=0.0
GO TO 440
430 CONTINUE
DELTX=X2-X1
DELTY=Y2-Y1
IF(DELTX)300,340,300
300 IF(DELTX)310,350,310
310 S=DELT Y/DELT X
   IF(S.GT.0.5.OR.S.LT.-0.5)GO TO 330
   DX=8.0*DELT X/ABS(DELT X)
   DY=S*DX
   GO TO 360
330 DX=4.0*DELT Y/ABS(DELT Y)
   DY=DX/S
   GO TO 360
340 IF(DELT X.EQ.0.0)DELT X=1.0
   DX=8.0*DELT X/ABS(DELT X)
   DY=0.0
   GO TO 360
350 DX=4.0*DELT Y/ABS(DELT Y)
   DX=0.0
360 IF(DX**2+DY**2.GT.0.25*(DELT X**2+DELT Y**2))GO TO 40
440 CALL VECTOR(X1+DX,Y1+DY,X2-DX,Y2-DY)
40 J=K
50 CONTINUE
   GO TO (200,51), LABEL
51 DO 20 I=1,NN
   X1=X(I)*SC+BD-XMIN
   Y1=Y(I)*SC+BD-YMIN
   CALL VECTOR(X1,Y1,X1,Y1)
20 CALL BOXNUM(I)
200 CALL VECTOR(0.0,0.0,0.0,0.0,0.0)
   GO TO 800
700 FPPLST=-1.0
   RETURN
800 FPPLST=0.0
   RETURN
900 FPPLST=1.0
   RETURN
END

**

FUNCTION FPASST(N)

C
C SEGMENT NAME..........................FPASST
C SEGMENT TYPE..........................FUNCTION
C SEGMENT LENGTH IN CARDS.................32
C DATE OF LAST ALTERATION...............20/07/76
C NUMBER OF LOCAL INTEGERS...............2
C NUMBER OF LOCAL REALS..................0
C
C FEHPOL ROUTINES CALLED..................FPTRAC
C                                          FPRLNM
C                                          NEXTIS
C FORTRAN LIBRARY ROUTINES CALLED...........IFIX
C OTHER LIBRARY ROUTINES CALLED............NONE
C CALLING ROUTINES.......................FPCMND
C COMMONS REFERENCED......................WKAREA
C                                          PARAMS
C
C REMARKS:
C THIS ROUTINE REPRESENTS THE FOLLOWING BNF RULES:
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPASST)
IF(NEXTIS(6HASSIGN, 6)) 700, 10, 900
10 IF(FPRLNM(0)) 700, 20, 700
20 I = IFIX(TM)
J = NEXTIS(2HT0, 2)
40 IF(NEXTIS(6HSOURCE, 6)) 700, 90, 50
50 IF(NEXTIS(8HDISPLAY, 8)) 700, 100, 60
60 IF(NEXTIS(8HLISTING, 8)) 700, 110, 70
70 IF(NEXTIS(6HREPLY, 6)) 700, 120, 75
75 IF(NEXTIS(8HKEEPFILE, 8)) 700, 80, 900
80 DA = I
'GO TO 800
90 SI = I
GO TO 800
100 DP = I
GO TO 800
110 LO = I
GO TO 800
120 RP = I
GO TO 800
700 FPASST = -1.0
RETURN
800 FPASST = 0.0
RETURN
900 FPASST = 1.0
RETURN
END

**

FUNCTION FPLEST(N)

SEGMEN NAME.......................... FPLEST
SEGMENT TYPE.......................... FUNCTION
SEGMENT LENGTH IN CARDS.................. 23
DATE OF LAST ALTERATION................... 20/07/76
NUMBER OF LOCAL INTEGERS.................. 3
NUMBER OF LOCAL REALS..................... 0

FEHPOL ROUTINES CALLED.................. FPTRAC
NEXTIS
FPLMNT
FPIDFR

FORTRAN LIBRARY ROUTINES CALLED......... FLOAT

OTHER LIBRARY ROUTINES CALLED........... COPY

CALLING ROUTINES........................ FPCMND
FUNCTION FPSEST(N)

COMMON/ WKAREA/ PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/ PARAMS/ SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HPFPLEST)

IF (NEXTIS (4HLET', 3)) 700, 10, 900
10 M=NE*KS+NE
   NK=NE*KS+1
   CALL COPY (1, PL(NK), 1, 2H:, 1)
   PL(M)=0.0
   KK=KP
15 IF (FPLMNT(0)) 700, 15, 20
20 IF (FPIDFR(-1)) 700, 30, 900
30 PL(KS NE)=FLOAT((KK-KP)/2)+0.5
GO TO 800
700 FPLEST=-1.0
RETURN
800 FPLEST=0.0
RETURN
900 FPLEST=1.0
RETURN
END

FUNCTION FPSEST(N)

COMMON/ WKAREA/ PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/ PARAMS/ SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HPFPLEST)

IF (NEXTIS (4HLET', 3)) 700, 10, 900
10 M=NE*KS+NE
   NK=NE*KS+1
   CALL COPY (1, PL(NK), 1, 2H:, 1)
   PL(M)=0.0
   KK=KP
15 IF (FPLMNT(0)) 700, 15, 20
20 IF (FPIDFR(-1)) 700, 30, 900
30 PL(KS NE)=FLOAT((KK-KP)/2)+0.5
GO TO 800
700 FPLEST=-1.0
RETURN
800 FPLEST=0.0
RETURN
900 FPLEST=1.0
RETURN
END
REMARKS:
THIS ROUTINE REPRESENTS THE BNF RULES:

<SET STATEMENT>::= SET <DEFINED TYPE NAME>=<REAL NUMBER> ON #
<SET STATEMENT>::= SET <DEFINED TYPE NAME>=<REAL NUMBER> ON #
<SET STATEMENT>::= SET <DEFINED TYPE NAME>=<REAL NUMBER> ON #

REAL XE(3), YE(3)
INTEGER CY(6,395), NT(395), JM(395), MJ(3950), BW, TN
REAL X(300), Y(300)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
NODE(INDEX, I) = PS-2*(INDEX+1)

10 CALL FPSKIP

20 CALL FPACCS(2, NMLST2, INDEX2, NMVALS)

30 IF(NEXTIS(2H= , 1)) 700, 30, 700

40 VALUE=TM

50 IF(NEXTIS(4HALL, 3)) 700, 340, 700

340 IF(NEXTIS(6HNODES, 5)) 700, 310, 700

310 DO 320 I=1, NN

230 PL(LL)=VALUE

GO TO 83

200 IF(NEXTIS(2H# , 1)) 700, 210, 700

210 IF(NPRLNM(0)) 700, 220, 700

220 NODEX=IFIX(TM)

225 IF(NODEX.GT.NN) GO TO 250

230 DO 320 I=1, NN

240 PL(LL)=-1111.0

240 PL(PT+1-INDEX2-NODEX)=VALUE

GO TO 83

250 CALL FPWARN(2)

300 IF(NEXTIS(2HON, 2)) 700, 50, 110

310 IF(NEXTIS(4HALL, 3)) 700, 340, 300

320 CALL FPWARN(2)

GO TO 225

60 CALL FPACCS(1, NMLST1, INDEX1, NMPTS)

L=1

CALL COMP(L, PL(NMLST1*NE+1), 1, 2H@ , 1)

IF(L.EQ.1) GO TO 90

L=1
CALL COMP(L, PL(NMNST1*NE +1), 1, 2H, 1)

********** NODAL SET **********

IF(NMVALS NE.0. AND. NMVALS NE. NN) GO TO 120
DO 80 I = 1, NN
LL=PT+1-INDEX2-I
IF(NMVALS .EQ. 0) PL(LL) = -1111.0
DO 70 J = 1, NMPTS
JJ=NODE(INDEX1, J)
IF(X(I).NE.PL(JJ).OR.Y(I).NE.PL(JJ-1)) GO TO 70
JM(J) = I
PL(LL) = VALUE
GO TO 80
70 CONTINUE
80 CONTINUE

IF(IE .NE. 6. OR. L .EQ. 0) GO TO 83
LZ=PT+1-INDEX2
DO 170 I = 1, NMPTS-1
I1=JM(I)
I2=JM(I+1)
DO 170 J = 1, NL
II=CY(1, J)
JJ=CY(2, J)
KK=CY(3, J)
IJ=CY(4, J)
JK=CY(5, J)
KI=CY(6, J)
IF(I1.EQ.II. AND. JJ.EQ.I2) PL(LZ-IJ) = VALUE
IF(JJ.EQ.II. AND. I1.EQ.I2) PL(LZ-IJ) = VALUE
IF(JJ.EQ.I1. AND. KK.EQ.I2) PL(LZ-JK) = VALUE
IF(KK.EQ.I2. AND. JJ.EQ.I1) PL(LZ-JK) = VALUE
IF(I1.EQ.I2. AND. KK.EQ.I1) PL(LZ-KI) = VALUE
IF(KK.EQ.I1. AND. JJ.EQ.I2) PL(LZ-KI) = VALUE
170 CONTINUE
83 PL(PS+NMLST2*NE+NE) = FLOAT(NN)
85 IF(NMVALS .EQ. 0) LP=LL-1
CALL FPTABL
87 IF(NEXTIS(2H, 1)) 700, 10, 800

********** ELEMENTAL SET **********

90 IF(NMVALS NE.0. AND. NMVALS NE. NL) GO TO 120
DO 100 I = 1, NL
LL=PT+1-INDEX2-I
DO 95 J = 1, 3
N=CY(J, I)
XE(J) = X(N)
95 YE(J) = Y(N)
XX=(XE(1)+XE(2)+XE(3))/3.0
YY=(YE(1)+YE(2)+YE(3))/3.0
IF(NMVALS .EQ. 0) PL(LL) = -1111.0
IF(FPINFNG(XX, YY, NMPTS, INDEX1). NE. 0.0) GO TO 100
PL(LL) = VALUE
100 CONTINUE
IF(NMVALS .EQ. 0) PL(PS+NMLST2*NE+NE) = FLOAT(NL)
GO TO 85

********** GLOBAL SET **********

110 KI=PS+NMLST2*NE+NE
XX=PL(KI)
PL(KI) = 0.0
LL = PT + 1 - INDEX2
115 LL = LL - 1
PL(LL) = VALUE
PL(KI) = PL(KI) + 1.0
IF(NMVALS .NE. 0 .AND. PL(KI) .GT. XX) GO TO 120
CALL FPSKIP
IF(FPRLNM(0))700,116,85
116 VALUE = TM
GO TO 115
120 CALL FPWARN(3)
WRITE(RP,130)IP
LS = LS - 1
GO TO 87
130 FORMAT(31H PARAMETER TYPE MISMATCH, COLUMN , I3/)
700 FPSEST = -1.0
RETURN
800 FPSEST = 0.0
RETURN
900 FPSEST = 1.0
RETURN
END

**

FUNCTION FPQUST(N)

**

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IF(NEXTIS(4HQUIT, 4))700, 10, 900
10 PT = 0
CALL FPSSTM
700 FPQUST = -1.0
RETURN
900 FPQUST = 1.0
RETURN
END

**

FUNCTION FPQUST(N)

C

SEGMENT NAME....................................................FPQUST
SEGMENT TYPE.....................................................FUNCTION
SEGMENT LENGTH IN CARDS.........................................11
DATE OF LAST ALTERATION.........................................20/07/76
NUMBER OF LOCAL INTEGERS.......................................0
NUMBER OF LOCAL REALS..........................................0

FFHPOL ROUTINES CALLED........................................NEXTIS
FPSSTM
FORTRAN LIBRARY ROUTINES CALLED...............................NONE

OTHER LIBRARY ROUTINES CALLED...............................NONE

CALLING ROUTINES..............................................FPCMMD

COMMANS REFERENCED............................................PARAMS

REMARKS:
THIS ROUTINE REPRESENTS THE FOLLOWING BNF RULE:
<QUIT STATEMENT> ::= QUIT

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IF(NEXTIS(4HQUIT, 4))700, 10, 900
10 PT = 0
CALL FPSSTM
700 FPQUST = -1.0
RETURN
900 FPQUST = 1.0
RETURN
END
FUNCTION FPGEST(N)

SEGMEN NAME .................. FPGEST  
SEGMEN TYPE .................. FUNCTION  
SEGMEN LENGTH IN CARDS. ........ 105  
DATE OF LAST ALTERATION ........ 20/07/76  
NUMBER OF LOCAL INTEGERS ........ 10  
NUMBER OF LOCAL REALS ........ 20  

FEHPOL ROUTINES CALLED .......... FPTRAC  
   NEXTIS  
   FPRLNM  
   FPSKIP  
   FPDTN  
   FPACC  
   FPENTR  
   FPINRG  
   FPTABL  

FORTRAN LIBRARY ROUTINES CALLED ... COS  
   FLOAT  
   SIN  
   SQRT  
   IFIX  

OTHER LIBRARY ROUTINES CALLED .... COPY  

CALLING ROUTINES ................ FPCMND  

COMMONS REFERENCED .............. WKAREA  
   PARAMS  
   MSAREA  

REMARKS:  
THIS ROUTINE REPRESENTS THE FOLLOWING BNF RULE:  
<GENERATE STATEMENT>::= GENERATE INTERNAL NODES FOR <DEFSI  
   GENERATE INTERNAL NODES FOR <DEFINED TYPENAME>  
<GENERATE STATEMENT>::= GENERATE <REAL NUMBER> INTERNAL NODES FOR <DEFINED T  

REAL DS(1)  
INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN  
REAL X(300),Y(300)  
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC  
REAL PL(1500),LN(20)  
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC  
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE  
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)  
EQUIVALENCE (DS(1),Y(300))  
S(X1,Y1)=X1**2+Y1**2  
PX(I)=0.41*COS(FLOAT(2*I))  
PY(I)=0.41*SIN(FLOAT(2*I))  
NODE(INDEX,I)=PS-2*(INDEX+I)  
CALL FPTRAC(6HFPGEST)  
IF(NEXTIS(8HGENERATE,8))700,50,900  
50 P=FPRLNM(0)  
   JP=NEXTIS(8HINTERNAL,8)  
IF(NEXTIS(6HNODES,5))700,60,700  
60 JP=NEXTIS(4HFOR,3)
CALL FPSKIP
JP=IP
IF(FPDTNM(0))700,70,700
70 CALL FPACC(1,NUMLST,INDEX,NMVALS)
IF(NUMLST.LT.0)GO TO 800
I=KS*NE+1
CALL COPY(1,PL(I),1,2H: ,1)
CALL FPENTR(1,IP)
KK=KP
K=NODE(INDEX,1)
J=NODE(INDEX,NMVALS)
P=S(PL(K)-PL(J),PL(K-1)-PL(J-1))
DS(NMVALS)=P
P=SQRT(P)
J=NODE(INDEX,1)
XMAX=PL(J)
YMAX=PL(J-1)
XMIN=XMAX
YMIN=YMAX
DSMIN=P
DO 10 I=1,NMVALS-1
J=NODE(INDEX,I)
K=J-2
DS(I)=S(PL(J-2)-PL(J),PL(J-3)-PL(J-1))
P=P+SQRT(DS(I))
IF(DS(I).LT.DSMIN)DSMIN=DS(I)
IF(PL(K).GT.XMAX)XMAX=PL(K)
IF(PL(K-1).GT.YMAX)YMAX=PL(K-1)
IF(PL(K).LT.XMIN)XMIN=PL(K)
10 IF(PL(K-1).LT.YMIN)YMIN=PL(K-1)
GRIDSZ=(P/FLOAT(NMVALS))
XMIN=XMIN+DSMIN
YMIN=YMIN+DSMIN
XMAX=XMAX-DSMIN
YMAX=YMAX-DSMIN
XD=XMAX-XMIN
YD=YMAX-YMIN
NX=IFIX(XD/GRIDSZ)
NY=IFIX(YD/GRIDSZ)
IF(NX.LE.1)NX=2
IF(NY.LE.1)NY=2
GRIDX=XD/FLOAT(NX)
GRIDY=YL/FLOAT(NY)
HGSX=GRIDX/2.0
HGSY=GRIDY/2.0
DO 21 I=1,NX
XC=FLOAT(I)*GRIDX-HGSX+XMIN
20 DO 30 J=1,NY
YC=FLOAT(J)*GRIDY-HGSY+YMIN
TOP=0.0
BOT=0.0
30 IF(D.EQ.0.0)GO TO 20
TOP=TOP+DS(K)/D
BOT=BOT+1.0/D
D = (TOP/BOT)
NZ = IFIX(GRIDX * GRIDY / D + 0.5)
IF(NZ .EQ. 0) NZ = 1
XCXC = X
YCYC = Y
NXY = NX * NY
40 PL(KP-1) = XCXC
   PL(KP-2) = YCYC
   IF(FPINRGS(XCXC, YCYC, NMVALS, INDEX) .EQ. 0.0) KP = KP-2
   NZ = NZ - 1
   IF(NZ .LE. 0) GO TO 20
   XCXC = XC + PX(NZ + NXY) * GRIDX
   YCYC = YC + PY(NZ + NXY) * GRIDY
   GO TO 40
20 CONTINUE
21 CONTINUE
   KK = KK - KP
   PL(KS * NE) = FLOAT(KK / 2)
   IF(KK .EQ. 0) KS = KS - 1
   CALL FPTABL
   GO TO 700
700 FPGEST = -1.0
   RETURN
800 FPGEST = 0.0
   RETURN
900 FPGEST = 1.0
   RETURN
END

FUNCTION FPKPST(N)

C SEGMENT NAME.......................... FPKPST
C SEGMENT TYPE.......................... FUNCTION
C SEGMENT LENGTH IN CARDS.............. 23
C DATE OF LAST ALTERATION.............. 20/07/76
C NUMBER OF LOCAL INTEGERS............. 0
C NUMBER OF LOCAL REALS................. 0
C FEHPOL Routines called.............. FPTRAC
C NEXTIS
C FORTRAN LIBRARY Routines called....... NONE
C OTHER LIBRARY Routines called........ NONE
C CALLING Routines..................... FPCMND
C COMMONS REFERENCED.................... WKAREA
C Params
C MSAREA

REMARKS:
THIS ROUTINE REPRESENTS THE FOLLOWING BNF RULE:
<KEEP STATEMENT>::= KEEP

INTEGER CY(6, 395), NT(395), JM(395), MJ(3950), BW, TN
REAL X(300), Y(300)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
FUNCTION FPUKST(N)

INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
CALL FPTRAC(6HFPKPST)
IF(NEXTIS(4HUNKEEP,4))700,10,900
10 REWIND DA
    WRITE (DA)
1 PL,KS,KP,LS,LP,LN,IP,TM,SK,TC,
2 SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE,
3 .CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
GO TO 800
700 FPKPST=-1.0
    RETURN
800 FPKPST=0.0
    RETURN
900 FPKPST=1.0
    RETURN
END

**FUNCTION FPUKST(N)**

"SEGMENT NAME" .................. FPUKST
"SEGMENT TYPE" .................. FUNCTION
"SEGMENT LENGTH IN CARDS" ........ 23
"DATE OF LAST ALTERATION" ........ 20/07/76
"NUMBER OF LOCAL INTEGERS" ........ 0
"NUMBER OF LOCAL REALS" ........ 0

"FEHPOL ROUTINES CALLED" ........ NONE
"FORTRAN LIBRARY ROUTINES CALLED" ........ NONE
"OTHER LIBRARY ROUTINES CALLED" ........ NONE
"CALLING ROUTINES" ........ FPCMND
"COMMONS REFERENCED" ........ WKAREA
                          PARAMS
                          MSAREA

"REMARKS:"
"THIS ROUTINE REPRESENTS THE BNF RULE:
<UNKEEP STATEMENT>::= UNKEEP "

"INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN"
"REAL X(300),Y(300)"
"INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC"
"REAL PL(1500),LN(20)"
"COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC"
"COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE"
"COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)"
"CALL FPTRAC(6HFPKPST)"
"IF(NEXTIS(6HUNKEEP,6))700,10,900"
10 REWIND DA
    READ (DA)
**FUNCTION FPNOST(N)**

INTEGER SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE,
       CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NE,TN,NP(9)
GO TO 800
700 FPUKST=-1.0
     RETURN
800 FPUKST=0.0
     RETURN
900 FPUKST=1.0
     RETURN
END

FUNCTION FPNOST(N)

C
SEGMENT NAME.................................FPNOST
SEGMENT TYPE...............................FUNCTION
SEGMENT LENGTH IN CARDS......................20
DATE OF LAST ALTERATION.....................20/07/76
NUMBER OF LOCAL INTEGERS....................2
NUMBER OF LOCAL REALS.......................0
FEHPOL ROUTINES CALLED.......................FPTRAC
FORTRAN LIBRARY ROUTINES CALLED.............NONE
OTHER LIBRARY ROUTINES CALLED..............COPY
CALLING ROUTINES............................FPCMND
COMMONS REFERENCED.........................WKAREA
REMARKS:
THIS ROUTINE IS A REPRESENTATION OF THE BNF RULE:
<NODES STATEMENT>::= NODES <DEFINITION>

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
CALL FPTRAC(6HFPNOST)
IF(N.EQ.1)GO TO 10
IF(NEXTIS(6HNODES ,5))700,10,900
10 M=NE*K+NE
   NK=NE*K+1
   CALL COPY(1,PL(NK),1,2H: ,1)
   PL(M)=0.0
   IF(FPDFIN(0))700,800,700
700 FPNOST=-1.0
     RETURN
800 FPNOST=0.0
     RETURN
900 FPNOST=1.0
     RETURN
END

**FUNCTION FPRGST(N)**
C C SEGMENT NAME..................FPRGST
C SEGMENT TYPE..................FUNCTION
C SEGMENT LENGTH IN CARDS........ 19
C DATE OF LAST ALTERATION........ 20/07/76
C NUMBER OF LOCAL INTEGERS......... 2
C NUMBER OF LOCAL REALS........... 0
C FEHPOL ROUTINES CALLED...........NEXTIS
C FORTRAN LIBRARY ROUTINES CALLED...FPDFIN
C OTHER LIBRARY ROUTINES CALLED.....NONE
C CALLING ROUTINES................FPCMND
C COMMONS REFERENCED................WKAREA
C PARAMS
C
C REMARKS:
C THIS ROUTINE IS A REPRESENTATION OF THE BNF RULE:
C (REGION STATEMENT)::= REGION <DEFINITION>
C
C INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
C REAL PL(1500),LN(20)
C COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
C COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
C CALL FPTRAC(6HPFPRGST)
C IF(NEXTIS(6HREGION,6))700,10,900
C 10 M=NE*KS+NE
C 20 NK=NE*KS+1
C 30 CALL COPY(1,PL(NK),1,2H§  ,1)
C 40 PL(M)=0.0
C 50 IF(FPDFIN(0))700,800,700
C 700 FPRGST=-1.0
C 800 FPRGST=0.0
C 900 FPRGST=1.0
C **
C FUNCTION FPCUST(N)
C
C SEGMENT NAME..................FPCUST
C SEGMENT TYPE..................FUNCTION
C SEGMENT LENGTH IN CARDS........ 19
C DATE OF LAST ALTERATION........ 20/07/76
C NUMBER OF LOCAL INTEGERS......... 2
C NUMBER OF LOCAL REALS........... 0
C FEHPOL ROUTINES CALLED...........NEXTIS
C FORTRAN LIBRARY ROUTINES CALLED...FPDFIN
C OTHER LIBRARY ROUTINES CALLED.....NONE
C CALLING ROUTINES................COPY
C COMMONS REFERENCED................WKAREA
C PARAMS
CALLING ROUTINES........................................FPCHND

COMMONS REFERENCED........................................WKAREA
 PARAMS

REMARKS:

THIS ROUTINE IS A REPRESENTATION OF THE FOLLOWING BNF
<CURVE STATEMENT>::= CURVE <DEFINITION>

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI-, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPCUST)
IF(NEXTIS(6HCURVE,5))700,10,900
10 M=NE*KS+NE
NK=NE*KS+1
CALL COPY(1,PL(NK),1,2H&,1)
PL(M)=0.0
IF(FPDFIN(0))700,800,700
700 FPCUST=-1.0
RETURN
800 FPCUST=0.0
RETURN
900 FPCUST=1.0
RETURN
END

FUNCTION FPLMNT(N)

SEGMENT NAME.................................................FPLMNT
SEGMENT TYPE.................................................FUNCTION
SEGMENT LENGTH IN CARDS..............................16
DATE OF LAST ALTERATION.........................20/07/76
NUMBER OF LOCAL INTEGERS.........................0
NUMBER OF LOCAL REALS...............................0

FEHPOL ROUTINES CALLED.................................FPTRAC
 FPCTPT
 FPDTNM
 FPSBSC

FORTRAN LIBRARY ROUTINES CALLED....................NONE

OTHER LIBRARY ROUTINES CALLED.......................NONE

CALLING ROUTINES........................................FPDFIN
 FPLEST

COMMONS REFERENCED........................................WKAREA
 PARAMS

REMARKS:

THIS ROUTINE REPRESENTS THE BNF RULE:
<ELEMENT>::= <CARTESIAN POINT>
<ELEMENT>::= <DEFINED TYPE NAME>
<ELEMENT>::= <DEFINED TYPE NAME> <SUBSCRIPT EXPRESSI

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
CALL FPTRAC(6HFPLMNT)
IF(FPCTPTT(0))700,800,10
10 IF(FPDTNM(0))700,20,900
20 IF(FPSBSC(0))700,800,800
700 FPLMNT=-1.0
RETURN
800 FPLMNT=0.0
RETURN
900 FPLMNT=1.0
RETURN
END

**
FUNCTION FPIDFR(N)

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
DATA IEXX/2HX /
CALL FPTRAC(6HFPPIDFR)
IF(N.EQ.-1)GO TO 5
IF(NEXTIS(2H:,1))700,10,900
5 IF(NEXTIS(2H=,1))700,8,900
8 I=NEXTIS(8HINTERNAL,8)
IF(NEXTIS(6HNODES,5))700,9,700
9 I=NEXTIS(4HFOR,3)
10 CALL FPSKIP
   SK=0
   JP=IP
   L=2
   IF(FPDTNM(L))700,20,700
20   SK=1
   L=IP-JP
   IF(N.EQ.-1)GO TO 25
   IF(TM)25,25,50
25   IF(L)30,30,40
30   CALL FPSTOP(32)
40   CALL FPENTR(1,JP)
   GO TO 800
700   FPIDFR=-1.0
   SK=1
   RETURN
50   CALL FPWARN(4)
   WRITE(RP,51)(IEEXX,I=1,L)
   CALL COPY(L,LN(1),JP,16HXXXXXXXXXXXXXXXX,1)
   GO TO 25
51   FORMAT(30H NAME NOT UNIQUE!! CHANGED TO ,20A1)
800   FPIDFR=0.0
   RETURN
900   FPIDFR=1.0
   RETURN
*
**
FUNCTION FPDTNM(I)
*
C
C SEGMENT NAME............................................FPDTNM
C SEGMENT TYPE..........................................FUNCTION
C SEGMENT LENGTH IN CARDS.................................43
C DATE OF LAST ALTERATION...............................20/07/75
C NUMBER OF LOCAL INTEGERS......................7
C NUMBER OF LOCAL REALS.................................0
C FEHPOL ROUTINES CALLED.................................FPTRAC
C..................................................FPLET
C..................................................FPDLET
C..................................................FPWARN
C FORTRAN LIBRARY ROUTINES CALLED..................FLOAT
C OTHER LIBRARY ROUTINES CALLED......................COMP
C CALLING ROUTINES........................................FPLMNT
C..................................................FPIDFR
C..................................................FPDLST
C..................................................FPGEST
C..................................................FPINST
C..................................................FPMBST
C..................................................FPPLST
C..................................................FPSEST
C..................................................FPWRST
C
C COMMONS REFERENCES.......................................WKAREA
C..................................................PARAMS
C
C REMARKS:
THIS ROUTINE IS A REPRESENTATION OF THE FOLLOWING BNF RULE:

<DEFINED TYPE NAME>::= <LETTER>
<DEFINED TYPE NAME>::= <DEFINED TYPE NAME><LETTER>
<DEFINED TYPE NAME>::= <DEFINED TYPE NAME><DIGIT>

IT SCANS FOR A NAME AND FINDS IT IN THE APPROPRIATE TABLE (I,0=SYMBOL TABLE, 1=PARAMETER TABLE) SETTING UP TM TO POINT TO IT, OR RETURN -1=NOT DEFINED.

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPDTNM)
IF(FPLETR(0))700, 10, 900
10 JP=IP-1
SK=0
20 IF(FPLETR(0))700,20,30
30 IF(FPDIGT(0))700,20,40
40 L=IP-JP
IF(L.GT-CN-1)L=CN-1
NUMSYM=KS
IF(I.EQ.1)NUMSYM=LS
IF(NUMSYM.EQ.0)GO TO 80
DO 50 M=1, NUMSYM
K=L
N=(M-1)*NE+1
IF(I.EQ.1)N=N+PS
ICHAR=2
IF(I.EQ.1)ICHAR=1
CALL COMP(K, LN(1), JP, PL(N), ICHAR)
IF(K.NE.L)GO TO 50
K=1
IF(L.NE-CN-1)CALL COMP(K, 2H, 1, PL(N), L+ICHAR)
IF(K.EQ.1)GO TO 60
50 CONTINUE
80 TM=-1.0
IF(I.GE.1)GO TO 70
CALL FPWARN(5)
GO TO 70
60 TM=FLOAT(M-1)
70 SK=1
GO TO 800
700 FPDTNM=-1.0
SK=1
RETURN
800 FPDTNM=0.0
RETURN
900 FPDTNM=1.0
RETURN
END

**
FUNCTION FPRLNM(I)

SEGMENT NAME..............FPRLNM
SEGMENT TYPE..............FUNCTION
SEGMENT LENGTH IN CARDS..............34
DATE OF LAST ALTERATION..............20/07/76
NUMBER OF LOCAL INTEGERS.......................... 1
NUMBER OF LOCAL REALS............................. 3

FEHPOL ROUTINES CALLED................................FPTRAC
NEXTIS
FPSKIP

FORTRAN LIBRARY ROUTINES CALLED.....................FLOAT

OTHER LIBRARY ROUTINES CALLED.........................NONE

CALLING ROUTINES........................................FPCTPT
FPINTP
FPGEST
FPSBSC
FPSEST
FPASST

COMMONS REFERENCED...................................WKAREA
PARAMS

REMARKS:
THIS ROUTINE SCANS FOR A REAL NUMBER, RETURNING
THE ACTUAL VALUE. A REAL NUMBER MUST START WITH
A DIGIT, THAT IS .002 IS INVALID.

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPRLNM)
SIGNR1.0  •
IF(NEXTIS(2H- , 1))700,70,80
80 CALL FPSKIP
SK=0
F=0.0
IF(FPDIGT(M))700,10,900
10 F=10.0*F+FLOAT(M)
IF(FPDIGT(M))700,10,20
20 IF(NEXTIS(2H. , 1))700,30,60
30 E=10.0
40 IF(FPDIGT(M))700,50,60
50 F=F+FLOAT(M)/E
E=10.0*E
GO TO 40
60 TM=F*SIGN
GO TO 800
70 SIGN=-1.0
GO TO 80
700 FPRLNM=-1.0
SK=1
RETURN
800 FPRLNM=0.0
SK=1
RETURN
900 FPRLNM=1.0
SK=1
RETURN
**FUNCTION FPCTPT(I)**

```
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/ WKAREA/ PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/ PARAMS/ SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
CALL FPTRAC(6HFPCTPT)
IF(FPRLNM(0))700, 10, 90 0
10 J=KP-1
   PL(J)=TM
   IF(NEXTIS(2H, 1))700, 20, 700
20 IF(FPRLNM(0) )700, 30,.700
   30 KP=KP-2
   CALL FPTABL
   PL(KP)=TM
   GO TO 800
700 FPCTPT= -1.0
   RETURN
800 FPCTPT= 0.0
   RETURN
900 FPCTPT= 1.0
   RETURN
END
```
FUNCTION FPDIGT(I)

INTEGER A,B,Z,L(1)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LH(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
DATA A/2HA /,B/2HB /,Z/2HZ /
CALL FPTRAC(6HFPLETR)

IDF=B-A
II=TC
TC=0
DO 10 I=A,Z,IDF
  L(1)=I
  IF(NEXTIS(L(1),1))700,800,10
10 CONTINUE
  TC=II
  IF(NEXTIS(2H-,1))700,800,900
700 FPLETR=-1.0
   TC=II
   RETURN
800 FPLETR=0.0
   TC=II
   RETURN
900 FPLETR=1.0
   RETURN
END
CALLING ROUTINES......................... FPRLNM
FPDTNM
COMMONS REFERENCED....................... WKAREA
PARAMS

REMARKS:
THIS ROUTINE SCANS FOR A DIGIT RETURNING ITS
ARGUMENT I AS THE VALUE OF THE DIGIT.

INTEGER ZERO,ONE,L(1)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
REAL PL(1500),LM(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
DATA ZERO/2H0 /,ONE/2H1 /,NINE/2H9 /
CALL FPTRACC(6HPDIGT)
IDF=ONE-ZERO
II=TC
TC=0
.I=0
.DO 10 J=ZERO,NINE,IDF
L(1)=J
IF(NEXTIS(L(1),1))700,800,10
10 I=I+1
I=0
GO TO 900
700 FPDTIGT=-1.0
TC=II
RETURN
800 FPDTIGT=0,0
TC=II
RETURN
900 FPDTIGT=1.0
TC=II
RETURN
END

**
FUNCTION NEXTIS(STRING,NUM)

SEGMENT NAME......................... NEXTIS
SEGMENT TYPE......................... FUNCTION
SEGMENT LENGTH IN CARDS............. 37
DATE OF LAST ALTERATION............ 20/07/76
NUMBER OF LOCAL INTEGERS........... 3
NUMBER OF LOCAL REALS.............. 0
FEHPOL ROUTINES CALLED................ FPTRAC
FORTTRAN LIBRARY ROUTINES CALLED.... ONE
OTHER LIBRARY ROUTINES CALLED....... COMP
CALLING ROUTINES...................... *****
COMMONS REFERENCED................... WKAREA
PARAMS
REMARKS:
THIS ROUTINE SCANS FOR 'STRING' OF LENGTH 'NUM'.
WHEN THE TRACE FLAG IS ON IT LISTS TO THE REPLY
DEVICE EACH STRING SOUGHT AS IT IS ENTERED TO-
GETHER WITH THE CURRENT RECORD. IT RECOGNIZES
THE SOUGHT STRING IF 2 OR MORE CHARACTERS ARE
RECOGNIZED (STATEMENT NUMBER 10) AND IF TRACE
IS ON NOTES THAT AN ALLOWABLE ABBREVIATION HAS
TAKEN PLACE ON THE REPLY DEVICE.

INTEGER BL
REAL STRING(4) .
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, IC, IS, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
DATA BL/2H  /
CALL FPTRAC(6HNEXTIS)
IF(SK)30,30,20
20 CALL FPSKIP
30 IF(TC.NE.1)GO TO 31
  IF(CR.EQ.4)WRITE(RP, 13) (LN(I), I=1, 20)
  IF(CR.EQ.6)WRITE(RP, 13) (LN(I), I=1, 14)
  IF(CR.EQ.8) WRITE(RP, 13) (LN(I), I=1, 10) ,
  WRITE(RP, 12) (BL, I=1, IP), STRING(1)
31 L=NUM
    CALL COMP(L, LN(1), IP, STRING(1), 1)
    IF(L.LT.NUM)GO TO 10
      IP=IP+L
      GO TO 800
10 IF(L.LT.2)GO TO 900
     IP=IP+L
     IF(TC.EQ.1)WRITE(RP, 11)
     GO TO 800
11 FORMAT(9H ABBREV!!)
12 FORMAT(1H ,80A1)
13 FORMAT(2H ,20A4)
14 FORMAT(2H ,13A6,A2),
15 FORMAT(2H ,10A8)
700 NEXTIS=-1
    RETURN
800 NEXTIS=0
    RETURN
900 NEXTIS=1
    RETURN
    END

**
SUBROUTINE FPSKIP

C
C SEGMENT NAME............................................FPSKIP
C SEGMENT TYPE............................................SUBROUTINE
C SEGMENT LENGTH IN CARDS.................................15
C DATE OF LAST ALTERATION.................................20/07/76
C NUMBER OF LOCAL INTEGERS...............................1
C NUMBER OF LOCAL REALS.................................0
SUBROUTINE FPNWCD

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), L(20)
COMMON/WKAREA/PL, KS, KP, IC, IS, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IP=IP-1
L=1
10 IP=IP+1
   IF(IP.GE.81)GO TO 30
20 CALL COMP(L, 2H, 1, LN(1), IP)
   IF(L)10, 40, 10
30 CALL FPNWCD
   GO TO 20
40 RETURN
END

**

REMARKS:
THIS ROUTINE SKIPS BLANKS UNTIL A NON-BLANK CHARACTER IS REACHED. IF NECESSARY, IT READS A NEW CARD UNTIL A NON-BLANK CHARACTER IS FOUND.

THIS ROUTINE READS AND LISTS A RECORD. IF THE FIRST TWO CHARACTERS OF THE RECORD ARE /* IT STOPS THE FEHPOL SYSTEM (MAINLY FOR BATCH MODE OPERATION). IF THE RECORD BEGINS WITH * IT IS A COMMENT AND SIMPLY LISTED AND ANOTHER ONE READ.
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, IC, IS, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
10 IF(CR.EQ.4)READ(SI,30)(LN(I),I=1,20)
   IF(CR.EQ.6)READ(SI,31)(LN(I),I=1,14)
   IF(CR.EQ.8)READ(SI,32)(LN(I),I=1,10)
40 IF(CR.EQ.4)WRITE(LO,50)(LN(I),I=1,20)
   IF(CR.EQ.6)WRITE(LO,51)(LN(I),I=1,14)
   IF(CR.EQ.8)WRITE(LO,52)(LN(I),I=1,10)
   IP=1
   SK=1
   L=2
   CALL COMP(L,2H/*,1,LN(1),1)
   IF(L.EQ.2)GO TO 20
   L=1
   CALL COMP(L,2H/*,1,LN(1),1)
   IF(L.NE.1)RETURN
   GO TO 10
20 STOP 00
30 FORMAT(20A4)
31 FORMAT(13A6, A2)
32 FORMAT(10A8)
50 FORMAT(1H ,20A4 )
51 FORMAT(1H ,13A6, A2 )
52 FORMAT(1H ,10A8 )
END

SUBROUTINE FPTRAC(STRING)

C         SUBROUTINE FPTRAC
C         SEGMENT NAME FPTRAC
C         SEGMENT TYPE SUBROUTINE
C         SEGMENT LENGTH IN CARDS 13
C         DATE OF LAST ALTERATION ' 20/07/76
C         NUMBER OF LOCAL INTEGERS 1
C         NUMBER OF LOCAL REALS 0
C         FEHPOL ROUTINES CALLED NONE
C         FORTRAN LIBRARY ROUTINES CALLED NONE
C         OTHER LIBRARY ROUTINES CALLED NONE
C         CALLING ROUTINES ********
C         COMMONS REFERENCED WKAREA PARAMS
C
C REMARKS:
C THIS ROUTINE WRITES OUT THE CURRENT RECORD WITH
C THE CHARACTER STRING 'STRING' AT THE CHARACTER
C LOCATION.
C
REAL STRING(2)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
SUBROUTINE FPCNTR(X1,Y1,X2,Y2,X3,Y3,XC,YC,RSQ)
C
C SEGMENT NAME..........................FPCNTR
C SEGMENT TYPE..........................SUBROUTINE
C SEGMENT LENGTH IN CARDS................22
C DATE OF LAST ALTERATION...............20/07/76
C NUMBER OF LOCAL INTEGERS..............0
C NUMBER OF LOCAL REALS................8
C
C FEHPOL ROUTINES CALLED................NONE
C FORTRAN LIBRARY ROUTINES CALLED........NONE
C OTHER LIBRARY ROUTINES CALLED.........NONE
C CALLING ROUTINES......................FPINTP
C........................................FPMSHP
C COMMONS REFERENCED....................NONE
C

REMARKS:
C THIS ROUTINE IS CALLED WITH THREE POINTS (X1,Y1)
C (X2,Y2),(X3,Y3). IT RETURNS (XC,YC)=THE CENTRE OF
C THE CIRCLE DEFINED BY THE THREE POINTS, AND RSQ=
C THE RADIUS SQUARED. IF THEY ARE COLINEAR, RSQ=A
C LARGE NUMBER.

X2SQ=X2*X2
Y2SQ=Y2*Y2
A=X2-X1
B=Y2-Y1
C=Y2SQ-Y1*Y1+X2SQ-X1*X1
D=X3-X2
E=Y3-Y2
F=Y3*X3-Y2SQ+X3*X3-X2SQ
X2SQ=D*B-E*A
IF(X2SQ)10,20,10
10 XC=(F*B-C*E)/X2SQ
YC=(D*C-F*A)/X2SQ
RSQ=(XC-X1)**2+(YC-Y1)**2
RETURN
20 RSQ=10000000000000.
RETURN
** END **

** FUNCTION FPAMNG(X,D,I) **

C SEGMENT NAME.................................FPAMNG
C SEGMENT TYPE.................................FUNCTION
C SEGMENT LENGTH IN CARDS.....................6
C DATE OF LAST ALTERATION....................20/07/76
C NUMBER OF LOCAL INTEGERS...................0
C NUMBER OF LOCAL REALS......................0
C
C FEHPOL ROUTINES CALLED......................NONE
C FORTRAN LIBRARY ROUTINES CALLED..............NONE
C OTHER LIBRARY ROUTINES CALLED...............NONE
C CALLING ROUTINES............................FPINTP
C COMMONS REFERENCED.........................NONE
C
C REMARKS:
C THIS ROUTINE GIVES THE INTERPOLATION FUNCTION OF
C AN ARBITRARY PLANAR CURVE BASED ON 4 POINT INTER-
C POLATION. OR THE END SEGMENTS, THE
C
C REAL X(30)
C FPAMNG=X(I-1)+(D+1.0)*((X(I)-X(I-1))+(D-1.0)*(-X(I-1)+3.0*X(I)-3.0*X(I+1)+X(I+2))/6.0))
RETURN
END

** SUBROUTINE FPCNPL(CONLEV,NUMLEV,INDX) **

C SEGMENT NAME.................................FPCNPL
C SEGMENT TYPE.................................SUBROUTINE
C SEGMENT LENGTH IN CARDS.....................104
C DATE OF LAST ALTERATION....................21/07/76
C NUMBER OF LOCAL INTEGERS...................13
C NUMBER OF LOCAL REALS......................6
C
C FEHPOL ROUTINES CALLED......................FPSTOP
C FORTRAN LIBRARY ROUTINES CALLED..............NONE
C OTHER LIBRARY ROUTINES CALLED...............NONE
C CALLING ROUTINES............................FPPLST
C COMMONS REFERENCED..........................PARAMS
C MSAREA
C
C REMARKS:
C THIS ROUTINE USES THE SUBCONNECTIVITY OF HIGHER
C ORDER ELEMENTS TO REDUCE THE CONTOUR PLOTTING
C OF THEM TO SEVERAL LINEAR TRIANGULAR ELEMENT
PLOTTING WHICH FPCNP3 CAN PERFORM. IT USES THE SHAPE FUNCTION OF THE ELEMENT TO FIND THE VALUES OF THE PARAMETER BEING PLOTTED AT EACH OF THE SUB-NODES (THAT IS THE POINTS AT WHICH EACH SUB-ELEMENT IN AN ELEMENT JOINS OTHERS).

ALGORITHM TO PLOT CONTOURS ON FINITE ELEMENT REGION USING THE SHAPE FUNCTION OF THE ELEMENTS

**DO FOR EVERY FINITE ELEMENT**

**DO FOR EVERY CONTOUR LEVEL**

**DO FOR NUMBER OF SUBNODES/ELEMENT**

* FIND X,Y COORDINATES OF THE SUBNODES USING THE TABLE OF HOMOGENOUS COORDINATES

* FIND THE VALUE OF THE FUNCTION AT EACH SUBNODE USING THE SHAPE FUNCTION FOR THIS TYPE OF ELEMENT

* CALL LINEAR THREE NODE TRI-ANGLE CONTOUR PLOTTING ROU-TINE FOR THE ITH SUB-ELEMENT USING SUBCONNECTIVITY TABLE
REAL CONLEV(NUMLEV),F(1),XX(15),YY(15),FF(15)
INTEGER CON6NT(3,16)
REAL H6NT(3,15)
INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW,TN
REAL X(300),Y(300)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CH,SK,TC
REAL PL(1500),LN(20)
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TH,SK,TC
COMMON/PARAMS/S5,DP,RP,LO,SV,DA,CR,PS,PT,CH,NE
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,HL,BW,NB,TN,NS(9)
EQUIVALENCE (Y(300),F(1))
DATA CON5NT(1,1),CON5NT(2,1),CON6NT(3,1) / 1,7,12/, 1
1 CON6NT(1,2),CON6NT(2,2),CON6NT(3,2) / 7,12,13/, 2
2 CON6NT(1,3),CON6NT(2,3),CON6NT(3,3) /13,12,6/, 3
3 CON6NT(1,4),CON6NT(2,4),CON6NT(3,4) / 6,13,15/, 4
4 CON6NT(1,5),CON6NT(2,5),CON6NT(3,5) /15,11,6/, 5
5 CON6NT(1,6),CON6NT(2,6),CON6NT(3,6) /11,15,10/, 6
6 CON6NT(1,7),CON6NT(2,7),CON6NT(3,7) /10,3,11/, 7
7 CON6NT(1,8),CON6NT(2,8),CON6NT(3,8) / 5,10,15/, 8
8 CON6NT(1,9),CON6NT(2,9),CON6NT(3,9) / 5,15,14/, 9
9 CON6NT(1,10),CON6NT(2,10),CON6NT(3,10) /14,15,13/, 1
1 CON6NT(1,11),CON6NT(2,11),CON6NT(3,11) /13,14,4/, 2
2 CON6NT(1,12),CON6NT(2,12),CON6NT(3,12) / 4,13,7/, 3
3 CON6NT(1,13),CON6NT(2,13),CON6NT(3,13) / 4,8,14/, 4
4 CON6NT(1,14),CON6NT(2,14),CON6NT(3,14) /14,8,9/, 5
5 CON6NT(1,15),CON6NT(2,15),CON6NT(3,15) / 9,14,5/, 6
6 CON6NT(1,16),CON6NT(2,16),CON6NT(3,16) / 2,8,9/ 10
DATA H6NT(1,1),H6NT(2,1),H6NT(3,1) /1.0,0.0,0.0/ 1
1 H6NT(1,2),H6NT(2,2),H6NT(3,2) / 0.0,1.0,0.0/ 2
2 H6NT(1,3),H6NT(2,3),H6NT(3,3) / 0.0,0.0,1.0/ 3
3 H6NT(1,4),H6NT(2,4),H6NT(3,4) / 0.5,0.5,0.0/ 4
4 H6NT(1,5),H6NT(2,5),H6NT(3,5) / 0.0,0.5,0.5/ 5
5 H6NT(1,6),H6NT(2,6),H6NT(3,6) / 0.5,0.0,0.5/ 6
6 H6NT(1,7),H6NT(2,7),H6NT(3,7) / 0.75,0.25,0.0/ 7
7 H6NT(1,8),H6NT(2,8),H6NT(3,8) / 0.25,0.75,0.0/ 8
8 H6NT(1,9),H6NT(2,9),H6NT(3,9) / 0.0,0.75,0.25/ 9
9 H6NT(1,10),H6NT(2,10),H6NT(3,10) / 0.0,0.25,0.75/ 1
1 H6NT(1,11),H6NT(2,11),H6NT(3,11) / 0.25,0.0,0.75/ 2
2 H6NT(1,12),H6NT(2,12),H6NT(3,12) / 0.75,0.0,0.25/ 3
3 H6NT(1,13),H6NT(2,13),H6NT(3,13) / 0.5,0.25,0.25/ 4
4 H6NT(1,14),H6NT(2,14),H6NT(3,14) / 0.25,0.5,0.25/ 5
5 H6NT(1,15),H6NT(2,15),H6NT(3,15) / 0.25,0.25,0.5/ 6
PHI6NT(EL1,EL2,EL3)= F(II)*EL1*(2.0*EL1-1.0) + 1
1 F(JJ)*EL2*(2.0*EL2-1.0) + 2
2 F(KK)*EL3*(2.0*EL3-1.0) + 3
3 F(IJ)*4.0*EL1*EL2 + 4
4 F(JK)*4.0*EL2*EL3 + 5
5 F(KI)*4.0*EL3*EL1
DO 10 I=1,NN
  10 F(I)=PL(PINDX-I)
   DO 20 I=1,NL
   II=CY(I,1)
   JJ=CY(I,2)
   KK=CY(I,3)
LL = CY(4, I)
IJ = LL
JK = CY(5, I)
KI = CY(6, I)
DO 30 J = 1, NUMLEV
C = CONLEV(J)
GO TO (100, 100, 300, 100, 100, 600, 100), IE
100 CALL FPSTOP(53)
300 CALL FPCNP3(X, Y, F, II, JJ, KK, C)
GO TO 30
600 DO 40 K = 1, 15
   XX(K) = X(II) * H6NT(1, K) + X(JJ) * H6NT(2, K) + X(KK) * H6NT(3, K)
   YY(K) = Y(II) * H6NT(1, K) + Y(JJ) * H6NT(2, K) + Y(KK) * H6NT(3, K)
40 FF(K) = P6WT(H6NT(1, K), H6NT(2, K), H6NT(3, K))
   DO 610 K = 1, 16
      III = CON6NT(1, K)
      JJJ = CON6NT(2, K)
      KKK = CON6NT(3, K)
610 CALL FPCNP3(XX, YY, FF, III, JJJ, KKK, C)
GO TO 30
30 CONTINUE
20 CONTINUE
RETURN.
END

* SUBROUTINE FPCNP3(X, Y, F, II, JJ, KK, C) *

SEGMENT NAME......................... FPCNP3
SEGMENT TYPE......................... SUBROUTINE
SEGMENT LENGTH IN CARDS.............. 34
DATE OF LAST ALTERATION.............. 21/07/76
NUMBER OF LOCAL INTEGERS............. 4
NUMBER OF LOCAL REALS.............. 15

FEHPOL ROUTINES CALLED............... NONE
FORTRAN LIBRARY ROUTINES CALLED...... AMIN1
OTHER LIBRARY ROUTINES CALLED........ VECTOR
CALLING ROUTINES..................... FPCNPPL
COMMONS REFERENCED.................. PLAREA

REMARKS:
THIS ROUTINE PLOTS A CONTOUR OF STATE VARIABLE F ON THE TRIANGLE (X(II), Y(II)) (X(JJ), Y(JJ)) (X(KK), Y(KK)) WITH CONTOUR LEVEL=C. IT ASSUMES A LINEAR LINTERPOLATION ON THE TRIANGLE.

REAL X(1), Y(1), F(1), XX(3), YY(3), FF(3), XXX(2), YYY(2)
INTEGER IP(3)
COMMON/PLAREA/BD, SC, PW
DATA IP(1), IP(2), IP(3)/2, 3, 1/
   XX(1) = X(II)
   XX(2) = X(JJ)
DO 10 J=1,3
K=IP(J)
YF=FF(K)-FF(J)
IF(YF)20,40,20
40 IF(C.NE.FF(K))GO TO 10

C * FOR THE CASE FF(J),FF(K)=C THE LOGIC OF THIS SEGMENT
C * IS NOT YET CORRECT. 31/08/76

10 CONTINUE
IF(I.NE.2)RETURN
30 XXX(1)=BD+SC*XXX(1)
XXX(2)=BD+SC*XXX(2)
YYY(1)=BD+SC*YYY(1)
YYY(2)=BD+SC*YYY(2)
CALL VECTOR(XXX(1),YYY(1),XXX(2),YYY(2))
RETURN
END

**

SUBROUTINE FPTABL

C C SEGMENT NAME............................................. FPTABL
C C SEGMENT TYPE........................................... SUBROUTINE
C C SEGMENT LENGTH IN CARDS................................. 9
C C DATE OF LAST ALTERATION......................... 21/07/76
C C NUMBER OF LOCAL INTEGERS............................. 0
C C NUMBER OF LOCAL REALS................................. 0
C C FEHPOL ROUTINES CALLED................................. FILE
C C FORTRAN LIBRARY ROUTINES CALLED.................... NONE
C C OTHER LIBRARY ROUTINES CALLED...................... NONE
C C CALLING ROUTINES.......................................****
C C COMMONS REFERENCED................................. WKAREA
C C PARAMS

XX(3)=X(KK)
YY(1)=Y(II)
YY(2)=Y(JJ)
YY(3)=Y(KK)
FF(1)=F(II)
FF(2)=F(JJ)
FF(3)=F(KK)
I=0
DO 10 J=1,3
K=IP(J)
YF=FF(K)-FF(J)
IF(YF)20,40,20

C FOR THE CASE FF(J),FF(K)=C THE LOGIC OF THIS SEGMENT
C IS NOT YET CORRECT. 31/08/76

10 CONTINUE
IF(I.NE.2)RETURN
30 XXX(1)=BD+SC*XXX(1)
XXX(2)=BD+SC*XXX(2)
YYY(1)=BD+SC*YYY(1)
YYY(2)=BD+SC*YYY(2)
CALL VECTOR(XXX(1),YYY(1),XXX(2),YYY(2))
RETURN
END

**

SUBROUTINE FPTABL

C C SEGMENT NAME............................................. FPTABL
C C SEGMENT TYPE........................................... SUBROUTINE
C C SEGMENT LENGTH IN CARDS................................. 9
C C DATE OF LAST ALTERATION......................... 21/07/76
C C NUMBER OF LOCAL INTEGERS............................. 0
C C NUMBER OF LOCAL REALS................................. 0
C C FEHPOL ROUTINES CALLED................................. FILE
C C FORTRAN LIBRARY ROUTINES CALLED.................... NONE
C C OTHER LIBRARY ROUTINES CALLED...................... NONE
C C CALLING ROUTINES.......................................****
C C COMMONS REFERENCED................................. WKAREA
C C PARAMS
REMARKS:
THIS ROUTINE CHECKS TO SEE IF EITHER THE SYMBOL TABLE OR THE PARAMETER TABLE HAS OVERFLOWED.

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SX, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SX, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
IF(KS*NE.LE.KP.AND.PS+LS*NE.LE.LP)RETURN
CALL FPWARN(13)
CALL FPSTOP(29)
END

**

FUNCTION FPSBSC(N)

SEGMENT NAME ........................................ FPSBSC
SEGMENT TYPE ........................................ FUNCTION
SEGMENT LENGTH IN CARDS ............................ 42
DATE OF LAST ALTERATION ............................ 21/07/75
NUMBER OF LOCAL INTEGERS ................................ 9
NUMBER OF LOCAL REALS .................................. 0

FEHPOL ROUTINES CALLED .............................. FPTRAC
FPACCS
NEXTIS
FPRLNM
FPSTABL

FORTRAN LIBRARY ROUTINES CALLED .................. IFIX

OTHER LIBRARY ROUTINES CALLED .................... NONE

CALLING ROUTINES ..................................... FPLMNT

COMMONS REFERENCED ................................. WKAREA
PARAMS

REMARKS:
THIS ROUTINE REPRESENTS THE BNF RULES:
<SUBSCRIPT EXPRESSION> ::= ( <REAL NUMBER> )
<SUBSCRIPT EXPRESSION> ::= ( <REAL NUMBER>-<REAL NUMBER> )

INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SX, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SX, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
NODE(INDEX, I)=P3-2*(INDEX+I)
CALL FPTRAC(6HFPSBSC)
10 CALL FPACCS(1,ITEMP,NLIST,M)
20 IF(NEXTIS(2H ,1))700,20,90
30 I=IFIX(TM)
40 IF(ITEMP.LT.O)GO TO 80
IEND=I
50 IF(NEXTIS(2H- ,1))700,40,50
60 IEND=IFIX(TM)
50 DO 70 J=I,IEND
   KP=KP-2
   KSN=NODE(NLIST,J)
   PL(KP)=PL(KSN-1)
   PL(KP+1)=PL(KSN)
   KSN=KSN+NE
   PL(KSN)=PL(KSN)+1.0
70 CONTINUE
80 IF(NEXTIS(2H),1)700,800,900
90 CALL FPACCS(1,ITEMP,NLIST,M).
   IF(ITEMP.LT.0) GOTO 800
   MM=NLIST+M
   DO 130 I=NLIST+1,MM
5   II=NODE(I,I)
6   KP=KP-2
   PL(KP)=PL(II-1)
   PL(KP+1)=PL(II)
   CALL FPTABL
   GO TO 900
700 FPSBSC=-1.0
   RETURN
800 FPSBSC=0.0
   RETURN
900 FPSBSC=1.0
   RETURN
  END
*
SUBROUTINE FPINTP(NODES,ITYPE)
*
SUBROUTINE
FPINTP
134
. . 21/07/76
10
71
FEHPOL ROUTINES CALLED......................... FPTRAC
FPSTOP
FPWARN
FPACCS
FPAMNG
FPCNTR
FPTABL
FPDLET
FORTRAN LIBRARY ROUTINES CALLED.................. FPACP
. IFIX
. SQRT
. ATAN2
. ABS
. COS
. SIN
 OTHER LIBRARY ROUTINES CALLED................... COPY
 CALLING ROUTINES............................... FPINTS
 COMMONS REFERENCED............................. WKAREA
 PARMS
REMARKS:
THIS ROUTINE PERFORMS EACH OF THE FOUR INTERPOLATION TYPES: SMOOTH, CIRCULAR, REGULAR AND LINEAR. IT USES THE OLD ENTRY FOR THE SYMBOL TO GENERATE A NEW SYMBOL WHICH IT GIVES THE SAME NAME AND DELETES THE OLD ONE.

REAL WA1(52), WA2(52), X(50), Y(50), WA3(52)
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/Sl, DP, RP, LS, SV, DA, CR, PS, PT, CN, NE
EQUIVALENCE (WA1(2), X(1)), (WA2(2), Y(1))
NODE(INDEX, I) = PS-2*(INDEX+I)
CALL FPTRAC(6HFPINTP)
IF(ITYPE.LT.1 OR ITYPE.GT.4) CALL FPSTOP(33)
IF(NODES.LE.1) CALL FPWARN(IO)
IF(NODES.GT.(KP-NE*KS)/2) CALL FPSTOP(U6)
FNODES = FLOAT(NODES)
LK = KP
SUM = 0.0
CALL FPACCS(1, ITEMP, NLST, M)
IF(ITEMP.LT.0) GO TO 120
IF(M.GE.NODES) GO TO 300
GO TO (90, 200, 100, 90), ITYPE
100 IF(M.GT.50) GO TO 300
FNO = FLOAT(M)-1.0
DO 150 I = 1, M
II = NODE(NLST, I)
X(I) = PL(II)
150 Y(I) = PL(II-1)
WA1(1) = X(1)
WA2(1) = Y(1)
WA1(M+2) = X(M)
WA2(M+2) = Y(M)
KP = KP-2
PL(KP+1) = X(1)
PL(KP) = Y(1)
DO 160 J = 2, NODES-1
FJ = FLOAT(J)-1.0
S = (FJ/FNODES)*FNO+1.0
I = IFIX(S)
D = S-FLOAT(I)
KP = KP-2
PL(KP+1) = FPAMNG(X, D, I)
PL(KP) = FPAMNG(Y, D, I)
160 KP = KP-2
PL(KP+1) = X(M)
PL(KP) = Y(M)
GO TO 110
200 IF(M.NE.3) GO TO 300
II = NODE(NLST, 0)-1
X1 = PL(II-1)
Y1 = PL(II-2)
X2 = PL(II-3)
Y2 = PL(II-4)
X3 = PL(II-5)
Y3=PL(II-6)
CALL FPCNTR(X1,Y1,X2,Y2,X3,Y3,XC,YC,R)
R=SQRT(R)
XT1=X1-XC
YT1=Y1-YC
XT2=X2-XC
YT2=Y2-YC
XT3=X3-XC
YT3=Y3-YC
A1=ATAN2(YT1,XT1)
A2=ATAN2(YT2,XT2)
A3=ATAN2(YT3,XT3)
A21=A2-A1
A31=A3-A1
IF(ABS(A31).LT.AB3(A21))A31 = (6.23318-A31)*A21/ABS(A21)
D=A31/(FNODES-1.0)
KP=KP-2
PL(KP+1)=X1
PL(KP)=Y1
DO 220 I=1,NODES-2
FI=FLOAT(I)
AI=A1+FI*D
XTI=R*COS(AI)
YTI=R*SIN(AI)
KP=KP-2
CALL FPTABL
PL(KP+1)=XTI+XC
PL(KP)=YTI+YC
IF(A2-A1).LT.0,110,240
240 KP=KP-2
PL(KP+1)=X3
PL(KP)=Y3
GO TO 110
90 DO 95 I=1,M
II=NODE(NLIST,I)
X(I)=PL(II)
95 Y(I)=PL(II-1)
NSLOTS=M-1
.R=0.0
FNEWNS=FNODES-FLOAT(M)
DO 50 I=1,NSLOTS
WA3(I)=SQRT((X(I+1)-X(I))**2+(Y(I+1)-Y(I))**2)
50 SUM=SUM+WA3(I)
DO 60 I=1,NSLOTS
WA3(I)=WA3(I)*FNEWNS/SUM
N=IFIX(WA3(I))
60 R=R+WA3(I)-FLOAT(N)
R=R/FLOAT(NSLOTS)
KP=KP-2
PL(KP+1)=X(I)
PL(KP)=Y(I)
NBTWN=IFIX(WA3(I)+0.99-R)
IF(NBTWN.LT.1)GO TO 70
FNBWNT=FLOAT(NBTWN+1)
IF(IATYPE.EQ.1)GO TO 65
WA1(I)=X(1)
$A_2(1) = Y(1)$
$X(M+1) = X(M)$
$Y(M+1) = Y(M)$

DO 69 J = 1, NBTWN
  FJ = FLOAT(J) / FNBTWN
  KP = KP - 2
  PL(KP+1) = FPAMNG(X, FJ, I)
  GO TO 70
69 DO 80 J = 1, NBTWN
  FJ = FLOAT(J) / FNBTWN
  KP = KP - 2
  KP = KP - 2
  PL(KP+1) = FPAMNG(Y, FJ, I)
  GO TO 70
70 CONTINUE

KP = KP - 2
PL(KP) = Y(NSLOTS+1)
PL(KP+1) = X(NSLOTS+1)
110 PL(KS*NE+NE) = FLOAT((LK-KP)/2)
CALL COPY(CN, PL(KS*NE+1), 1, PL(ITEMP*NE+1), 1)
KS = KS + 1
CALL FPDLET(ITEMP)
120 RETURN

**

SUBROUTINE FPMESH(S1, S2, S3, AA, CY, NS, NR, X, Y, M1, M2, NC,  
1 NL, NB, NP)

C
C SEGMENT NAME .............................................. FPMESH
C SEGMENT LENGTH IN CARDS .................................. 126
C DATE OF LAST ALTERATION ................................. 21/07/76
C NUMBER OF LOCAL INTEGERS ................................ 18
C NUMBER OF LOCAL REALS .................................... 13
C
C FEHPOL ROUTINES CALLED ..................................... FPSSOL
C
C FORTRAN LIBRARY ROUTINES CALLED .......................... SQRT
C OTHER LIBRARY ROUTINES CALLED ............................ NONE
C CALLING ROUTINES .......................................... FPCNCT
C COMMONS REFERENCED ........................................ NONE
C
C REMARKS:
C THIS ROUTINE PERFORMS THE CONNECTION OF A FINITE
C ELEMENT MESH GIVEN THE LOCATION OF ALL THE NODES
C AND IDENTIFICATION OF THE BOUNDARY NODES.
C
C ARGUMENTS:
C S1, S2, S3 = WORKAREA FOR STACKS, LENGTH AT LEAST M1
C AA = WORKAREA, LENGTH AT LEAST M2
C NS = WORKAREA, LENGTH AT LEAST M2

RETURN
RETURN
CALL FPWARN(6)
RETURN
END

**
NS, NR = WORK AREAS, LENGTH AT LEAST M2
X, Y = NODE COORDINATES, DIMENSION NC
M1 = MAXIMUM NUMBER OF ELEMENTS ALLOWED
M2 = MAX NUMBER OF EQUALLY ATTRACTIVE CONNECTIONS FOR ANY BASELINE. FOR ANY REASONABLE MESH M2 = 20
NC, NL, NB = NUM OF CORNER POINTS, ELEMS, BOUNDARIES
NP(I) = NUMBER OF NODES ON THE ITH BOUNDARY

FLOWCHART OF TRIANGULAR MESH GENERATION ALGORITHM

***************
* PICK BASELINE I-J *
* ON THE BOUNDARY AND *
* POINT N BEHIND IT. *
***************

**************
V
**************

***************
FIND NS(M), M = 1, IR
EQUALLY MOST SUITABLE FOR I-J ON SIDE
OPPOSITE N AND NOT CROSSING ANY BOUNDARY
***************

***************
REORDER NS(M) TO PIVOT ON J
***************

***************
FOR M = 1, IR:
ADD ELEM I-J-NS(M)
STACK SIDE I-NS(M)
WITH BEHIND NODE J
N = I, I = NS(M)
***************

***************
I-J A Y
BOUNDARY SEGMENT?
***************

N
V

IS THE STACK EMPTY?

I-J IN TWO ELEMENTS?

DONE
INTEGER S1(M1),S2(M1),S3(M1),CY(6,395),NS(M2),NR(M2),NP(1)
REAL X(NC),Y(NC),AA(M2)
D(X1,Y1,X2,Y2)=SQRT((X2-X1)**2+(Y2-Y1)**2)
SP=0
XG=0
I=NP(1)
J=1
N=1
RN=10000000000000.
GO TO 35
1000 RN=10000000000000.
C *************************************#*************
C * SEE IF I-J IS ON A BOUNDARY. IF YES, GO TO 120. *
C *************************************  »••••••••••••••
IT=0 ,
IL=1
DO 1020 MM=1,NB
IT=IT+NP(MM)
IF(IL.GT.J)GO TO 1020
IF(IL.GT.I)GO TO 1020
IF(J.GT.IT)GO TO 1020
IF(I.GT.IT)GO TO 1020
IF(ABS(I-J).EQ.1)GO TO 120
IF(ABS(I-J).EQ.NP(MM)-1)GO TO 120
1020 IL=IT+1
NU=0
DO 40 L=1,NG
II=CY(1,L)
JJ=CY(2,L)
KK=CY(3,L)
IF(I.NE.II.AND.I.NE.JJ.AND.I.NE.KK)GO TO 40
IF(J.NE.II.AND.J.NE.JJ.AND.J.NE.KK)GO TO 40
NU=NU+1
IF(NU.GE.2)GO TO 120
40 CONTINUE
35 XI=X(I)
XJ=X(J)
XN=X(N)
YI=Y(I)
YJ=Y(J)
YN=Y(N)
DSQ=((XJ-XI)**2+(YJ-YI)**2)/4.0
DO 60 L=1,NC
IF(L.EQ.I)GO TO 60
IF(L.EQ.J)GO TO 60
XL=X(L)
YL=Y(L)
IF(FPSSOL(XI,YI,XJ,YJ,XL,YL,XN,YN))80,80,60
CALL FPCNTR(XI,YI,XJ,YJ,XL,YL,XC,YC,RSQ)
R=RSQ-DSQ
IF(FPSSOL(XI,YI,XJ,YJ,XC,YC,XL,YL))314,314,313
R=-R
IF(R.GT.RN)GO TO 60
C************************************************
C * CHECK TO SEE IF SEGMENT I-L CROSSES BOUNDARY. *
C * IF YES, GO TO 50 *
C************************************************
IT=0
IL=1
DO 1030 MM=1,N5
  IT=IT+NP(MM)
  DO 95 NO=IL,IT
    N1=NO+1
    IF(NO.EQ.IT)N1=IL
    IF(NO.EQ.I)GO TO 95
    IF(NO.EQ.L)GO TO 95
    IF(N1.EQ.I)GO TO 95
    IF(N1.EQ.L)GO TO 95
    IF(FPSSOL(XI,YI,XL,YL,X(NO),Y(NO),X(N1),Y(N1)))95,96,95
  95 CONTINUE
C************************************************
C * CHECK TO SEE IF SEGMENT J-L CROSSES BOUNDARY. *
C * IF YES, GO TO 50 *
C************************************************
DO 97 NO=IL,IT
  N1=NO+1
  IF(NO.EQ.IT)N1=IL
  IF(NO.EQ.J)GO TO 97
  IF(NO.EQ.L)GO TO 97
  IF(N1.EQ.J)GO TO 97
  IF(N1.EQ.L)GO TO 97
  IF(FPSSOL(XJ,YJ,XL,YL,X(NO),Y(NO),X(N1),Y(N1)))97,98,97
96 IF(FPSSOL(X(NO),Y(NO),X(N1),Y(N1),XI,YI,XL,YL))95,60,97
95 CONTINUE
DO 1030 IL=IT+1
  IF(R.LT.RN)GO TO 90
  IR=IR+1
  NS(IR)=L
  GO TO 60
90 CONTINUE
RN=R
IR=1
NS(IR)=L
60 CONTINUE
IF(IR.GT.M2)GO TO 920
DO 30 II=1,IR
  IO=NS(II)
  AA(II)=D(XJ,YJ,X(IO),Y(IO))-D(XI,YI,X(IO),Y(IO))
30  CONTINUE
DO 25 II=1,IR
  IO=0
  DO 20 JJ=1,IR
    IF(AA(JJ).GE.AA(II))IO=IO+1
  20 CONTINUE
25 NR(IO)=NS(II)
the nodes and initializes the routines to generate
the nodes and initializes the routines to generate

it initializes the renumbering scheme, reorders
performs checking of the mesh. if it is alight

this routine sets up the call to pmesh and

remarks:

marks

parameters

common area referenced.

calling routines.

end

none.

other library routines called.

none.

portran library routines called.

philip routines called.

number of local reals.

number of local integers.

date of last alteration.

segment length in cards.

segment name. 

segment name. 

subroutine pmpc1

end

return

If(sp) (sp) 920, 1000, 1000

If(sp) = sp - 1

N = (sp) (sp)

J = (sp) (sp)

120 I = sp (sp)

*****************************************************************************

* unstack baseline and behind node (i,j) and (n)

*****************************************************************************

Go to 1000

10 If (n) (n)

i = n

(i'j') = (n)

j = (n)

1

*****************************************************************************

add new element i-j-nr(ii) to connectivity table

*****************************************************************************

s3 = sp (j)

s2 = sp (ii)

s1 = sp (j)

sp = sp + 1

ng = ng + 1

Do 10 ii = 1, 1
SIX NODED CONNECTIVITY IF REQUESTED. IT ALSO MAKES SURE THAT THE ELEMENTS ARE ALL ENTERED IN COUNTERCLOCKWISE ORDER SINCE MANY SOLUTION PROGRAMS REQUIRE THIS.

REAL X(300),Y(300),AA(20)
INTEGER CY(6,395),MJ(3950),NT(395),JM(395),SP,BW,TN
REAL PL(1500),LN(20)
INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT,CN,SK,TC
COMMON/WKAREA/PL,KS,KP,LS,LP,LN,IP,TM,SK,TC
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
COMMON/MSAREA/CY,NT,JX,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NT(9)
IF(IE.LT.3.OR.IE.GT.6)CALL FPSTOP(34)
IF(NC.GT.300)CALL FPSTOP(39)
NI=NC-TN
MX=395
MY=20
CALL FPMESH(NT(1),JM(1),MJ(1),AA,CY,MJ(MX+1),MJ(MX+1+MY),
 X,Y,MX,MY,NC,NL,BW,NN,TN,NT(9))
IF(NL.EQ.(NC-2+NB)*2-TN)GO TO 930
CALL FPWARN(7)
IE=3
RETURN
930 IF(NC.GT.300)CALL FPSTOP(39)
BW=0
CALL FPNUM
IF(BW.EQ.0)RETURN
NN=NC
IF(IE.EQ.6)CALL FPGSN6
CALL FPORG
DO 499 I=1,NL
II=CY(1,I)
JJ=CY(2,I)
K=CY(3,I)
A2=X(KK)-X(II)
A3=X(JJ)-X(II)
B2=Y(KK)-Y(II)
B3=Y(JJ)-Y(II)
IF(A3*B2-B3*A2)510,499,499
510 II=CY(3,I)
CY(3,I)=CY(1,I)
CY(1,I)=II
IF(IE.EQ.3)GO TO 499
II=CY(4,I)
CY(4,I)=CY(5,I)
CY(5,I)=II
499 CONTINUE
RETURN
END

**

SUBROUTINE FPGEN6
C
C SEGMENT NAME:.................FPGEN6
C SEGMENT TYPE:..................SUBROUTINE
**SEGMENT LENGTH IN CARDS**: 49

**DATE OF LAST ALTERATION**: 21/07/76

**NUMBER OF LOCAL INTEGERS**: 12

**NUMBER OF LOCAL REALS**: 0

**FEHPOL ROUTINES CALLED**: FPSTOP

**FORTRAN LIBRARY ROUTINES CALLED**: MAXO

**OTHER LIBRARY ROUTINES CALLED**: NONE

**CALLING ROUTINES**: FP CNCT

**COMMONS REFERENCED**: MSAREA

**REMARKS**: 
This routine takes optimally numbered three nodded connectivity and produces optimally numbered six nodded connectivity. It also enters the side nodes into the coordinates table and recalculates the semi-bandwidth.

**ALGORITHM TO GENERATE SIDE NODED CONNECTIVITY**

```
* ************************************************************
* BANDWIDTH=0, NUM=0 *
* ************************************************************

V

* FOR L=1 TO NUMBER OF CORNER POINTS *
*  "FIND NODE L " *
*  "V "
*  " "
*  "0 "
*  " NODES Y "
*  " CONNECTED "
*  " TO L? "
*  " "
*  " N "
*  "K K=NUM=NUM+1 GIVE "
*  " L NEW NUMBER=NUM "
*  " "
*  " V "

* FOR J=1 TO NUMB. OF NODES CONNECTED TO L *
*  "M= NODE NUMBER OF "
*  "JTH NODE CONNECTED TO NODE L "
*  "V "
```
REAL X(300), Y(300)
INTEGER CY(6, 395), NT(395), JM(395), MJ(3950), BW, TN
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
IF(NC+(NL*3+TN)/2 .GT. 300) CALL FPSTOP(37)

BW=0
NUM=0
INDEX=NC
DO 450 L=1,NC
DO 403 J=1,NC
 1 = J
  IF(NT(J) .EQ. DO) GO TO 404
  403 CONTINUE
  CALL FPSTOP(38)
  404 NUM=NUM+1
  KKK=NUM
  MM=JM(I)
  IF(MM.EQ.0) GO TO 450
  JM(I)=NUM
  DO 440 J=1,MM
  M=MJ(J+(I-1)*10)
  IF(NT(M).LT.DO) GO TO 440
  NUM=NUM+1
  DO 430 K=1,NL
  II=CY(1,K)
  JJ=CY(2,K)
  KK=CY(3,K)
  IF(II.NE.I.AND.JJ.NE.I.AND.KK.NE.I) GO TO 430
  IF((II.NE.I.AND.JJ.NE.I.AND.KK.NE.I) .OR. (II.EQ.I.AND.JJ.EQ.I)) GO TO 420
  IF((JJ.EQ.M.AND.KK.EQ.I).OR.(JJ.EQ.I.AND.KK.EQ.M)) GO TO 410
  CY(6,K)=NUM
  GO TO 430
  410 CY(5,K)=NUM
GO TO 430
420 CY(4,K)=NUM
430 CONTINUE
INDEX=INDEX+1
X(INDEX)=(X(I)+X(M))/2.0
Y(INDEX)=(Y(I)+Y(M))/2.0
NT(INDEX)=NUM
BW=MAX0(BW,NUM-KKK+1)
440 CONTINUE
450 CONTINUE
NN=NUM
DO 460 I=1,NC
460 NT(I)=JM(I)
RETURN
END

**
SUBROUTINE FPRORG

C
C SEGMENT NAME.................... FPRORG
C SEGMENT TYPE................... SUBROUTINE
C SEGMENT LENGTH IN CARDS........ 20
C DATE OF LAST ALTERATION....... 21/07/76
C NUMBER OF LOCAL INTEGERS....... 2
C NUMBER OF LOCAL REALS.......... 0
C
C FEHPOL ROUTINES CALLED......... NONE
C FORTRAN LIBRARY ROUTINES CALLED..... NONE
C OTHER LIBRARY ROUTINES CALLED..... NONE
C CALLING ROUTINES............... FPCNCT
C COMMONS REFERENCED........... MSAREA
C
C REMARKS:
C THIS ROUTINE TAKES THE RENUMBERING TABLE GENERATED
C BY COLLINS RENUMBERING ALGORITHM AND REORDERS THE
C NODES TO CONFORM.
C
REAL X(300),Y(300)
INTEGER CY(6,395),NT(395),JM(395),MJ(3950),BW, TN
COMMON/MSAREA/CY, NT, JM, X, Y, MJ, IE, NN, NC, NL, BW, NB, TN, NP(9)
DO 480 J=1,NL
480 CY(I,J)=NT(CY(I,J))
DO 497 I=1,NN
IF(I.NE.NT(J))GO TO 496
Y(2*NN+I)=X(2)
Y(NN+I)=Y(J)
GOTO 497
496 CONTINUE
497 CONTINUE
DO 498 I=1,NN
X(I)=Y(I+2*NN)
FUNCTION FPSSOL(XI,YI,XJ,YJ,XL,YL,XN,YN)

A=XJ-XI
IF(A)70,50,70
50 IF(XJ.LT.XL.AND.XJ.GE.XN)GO TO 80
IF(XJ.GT.XL.AND.XJ.LE.XN)GO TO 80
GO TO 60
70 S=(YJ-YI)/A
A=S*(XL-XJ)
B=S*(XN-XJ)
C=YL-YJ
D=YN-YJ
IF(C.LT.A.AND.D.GE.B)GO TO 80
IF(C.GT.A.AND.D.LE.B)GO TO 80
60 FPSSOL=1.0
RETURN
80 FPSSOL=0.0
RETURN
END

SUBROUTINE FPRNUM

INTEGER SI,DP,RP,SV(9),DA,CR,PS,PT, CN, SK, TC
INTEGER CY(6,395),NT(395),JM(395),MJ(3950),NEWJT(300)
1,JOINT(300),BW,TN
REAL X(300),Y(300)
COMMON/MSAREA/CY,NT,JM,X,Y,MJ,IE,NN,NC,NL,BW,NB,TN,NP(9)
COMMON/PARAMS/SI,DP,RP,LO,SV,DA,CR,PS,PT,CN,NE
ID=NC
IF(NL.EQ.0)RETURN
DO 10 J=1,NC
10 JM(J)=0
DO 60 J=1,NL
   DO 50 I=1,3
      JNTI=CY(I,J)
      IF(JNTI)15,60,15
15 JSUB=(JNTI-1)*10
   DO 40 II=1,3
      IF(II.EQ.I)GO TO 40
      JJT=CY(II,J)
      IF(JJT)16,50,16
16 MEM1=JM(JNTI)
      IF(MEM1)17,30,17
17 DO 20 III=1,MEM1
      IF(MJ(JSUB+III).EQ.JJT)GO TO 40
   20 CONTINUE
50 CONTINUE
60 CONTINUE
**#*****#*******SMOOTH MESH IF REQUIRED*************
I=NEXTIS(2H,1)
IF(NEXTIS(6HSMOOTH,6))330,340,320
330 STOP 66
340 CONTINUE
   DO 300 I=NB+1,NC
      IF(MJ(3951+NB-I).EQ.-1)GO TO 300
      XSUM=0.0
      YSUM=0.0
   DO 310 J=1,JM(I)
      XSUM=XSUM+X(MJ((1-1)*10+J))
310 YSUM=YSUM+Y(MJ((I-1)*10+J))
   X(I)=XSUM/FLOAT(JM(I))
   Y(I)=YSUM/FLOAT(JM(I))
300 CONTINUE
320 CONTINUE
   NN=NC
   DO 950 I=1,NC
      IF(JM(I).EQ.0)NN=NN-1
      IF(JM(I).LE.10)GO TO 950
      CALL FPWARN(8)
      NN=NC
      IE=3
   RETURN
950 CONTINUE
   BW=ID
   MINMAX=ID
   DO 260 IK=1,NN
**SUBROUTINE FPWARN(N)**

INTEGER BL, ARO
INTEGER SI, DP, RP, SV(9), DA, CR, PS, PT, CN, SK, TC
REAL PL(1500), LN(20)
COMMON/WKAREA/PL, KS, KP, LS, LP, LN, IP, TM, SK, TC
COMMON/PARAMS/SI, DP, RP, LO, SV, DA, CR, PS, PT, CN, NE
DATA BL/2H, ARO/2H /

IF(N.EQ. 1)WRITE(RP, 1)
IF(N.EQ. 2)WRITE(RP, 2)
IF(N.EQ. 3)WRITE(RP, 3)
IF(N.EQ. 4)WRITE(RP, 4)
IF(N.EQ. 5)WRITE(RP, 5)
IF(N.EQ. 6)WRITE(RP, 6)
IF(N.EQ. 7)WRITE(RP, 7)
IF(N.EQ. 8)WRITE(RP, 8)
IF(N.EQ. 9)WRITE(RP, 9)
IF(N.EQ. 10)WRITE(RP, 10)
IF(N.EQ. 11)WRITE(RP, 11)
IF(N.EQ. 12)WRITE(RP, 12)
IF(N.EQ. 13)WRITE(RP, 13)
IF(N.EQ. 14)WRITE(RP, 14)
IF(CR.EQ. 4)WRITE(RP, 21)(LN(I), I=1, 20), (BL, I=1, IP), ARO
IF(CR.EQ. 6)WRITE(RP, 22)(LN(I), I=1, 14), (BL, I=1, IP), ARO
IF(CR.EQ. 8)WRITE(RP, 23)(LN(I), I=1, 10), (BL, I=1, IP), ARO
RETURN
21 FORMAT(2H, 20A4/1H, 80A1)
22 FORMAT(2H, 13A6, A2/1H, 80A1)
23 FORMAT(2H, 10A8/1H, 80A1)
1 FORMAT(15H, SYNTAX ERROR!)
2 FORMAT(24H, NODE NUMBER TOO LARGE!)
3 FORMAT(23H, PARAMETER TYPE ERROR!)
4 FORMAT(18H, NAME NOT UNIQUE!)
5 FORMAT(17H, UNDEFINED NAME!)
6 FORMAT(24H, WRONG NUMBER OF NODES!)
7 FORMAT(37H, UNUSUAL MESH! OPTIMIZATION OMITTED!)
8 FORMAT(38H, MORE THAN TEN CONNECTIONS AT A NODE!)
9 FORMAT(26H, TOO FEW BOUNDARY NODES!)
10 FORMAT(17H, TOO FEW POINTS!)
11 FORMAT(31H, CONNECTIVITY TABLE IS EMPTY!!)
12 FORMAT(22H, SYMBOL TABLE EMPTY!!)
13 FORMAT(22H, SYMBOL TABLE CRASH!!)
14 FORMAT(38H, TOO FEW NODES ON INTERNAL BOUNDARY!!)
END
C
**********END OF SOURCE******************
APPENDIX B

FEHPOP User's Guide
The FEHPOL System

by

J. M. Nelson

University of Southampton
Highfield, Southampton

May 1976
<table>
<thead>
<tr>
<th>Introduction</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1</td>
<td>Simple Use of FEHPOL</td>
</tr>
<tr>
<td>Part 2</td>
<td>The FEHPOL Commands</td>
</tr>
<tr>
<td>Part 3</td>
<td>The FEHPOL Abort Messages</td>
</tr>
</tbody>
</table>
INTRODUCTION

FEHPOL is a computer system written at the University of Southampton to allow users to build finite element models of coastal and estuarial hydraulics. The aim has been to provide an easy to understand interactive interface between a finite element solution program and the civil engineer. FEHPOL offers several facilities and overcomes many of the failings of earlier systems of this nature. It considerably reduces the time required to model a given physical problem.
Part 1
Simple Use of FEHPOL

This section describes how to define a region and generate a finite element mesh on it. The description is intuitive, relying on a basic knowledge of the finite element method.

Consider a square with corners located (in cartesian coordinates) at \([0,0]\) \([1,0]\) \([1,1]\) and \([0,1]\). To generate a simple finite element mesh on this square and display its connectivity the following FEHPOL command sequence could be issued:

```
REGION 0,0 1,0 1,1 0,1; SQUARE
MESH SQUARE WITH THREE NODDED TRIANGLES
WRITE CONNECTIVITY
```

The latter of the FEHPOL statements would cause the following to be printed out:

```
CONNECTIVITY TABLE
  1  1  3  2  2  3  2  4
NODAL COORDINATES
  1 0.0000 1.0000 2 0.0000 0.0000
  3 1.0000 1.0000 4 1.0000 0.0000
BANDWIDTH= 3
```

The numbering scheme has been chosen which minimizes the bandwidth of the resulting matrix equation. To plot the mesh which has been generated the user could issue the command:

```
PLOT MESH or PLOT AND LABEL MESH
```
In the second case, the nodes are labelled as in Figure 1.

![Figure 1](image)

In addition he could type:

- WRITE SQUARE
- PLOT SQUARE

which would output the following and Figure 2 respectively.

<table>
<thead>
<tr>
<th>@SQUARE</th>
<th>X-COORD</th>
<th>Y-COORD</th>
<th>X-COORD</th>
<th>Y-COORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0000</td>
<td>0.0000</td>
<td>2</td>
<td>0.0000</td>
</tr>
<tr>
<td>3</td>
<td>1.0000</td>
<td>1.0000</td>
<td>4</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

![Figure 2](image)
If the user wished to generate a mesh with two internal nodes in it, he could type the following command sequence:

\[
\text{REGION 0,0 1,0 1,1 0,1;SQUARE}
\]
\[
\text{LET 0.3,0.5 0.7,0.5=INTERNAL NODES FOR SQUARE}
\]
\[
\text{MESH SQUARE WITH TRED NODED TRIANGLES}
\]
\[
\text{PLOT AND LABEL MESH}
\]

In this case, a plot like Figure 3 would be produced.

![Figure 3](image)

The user may wish to generate more nodes along the sides of the square. To do this he must define each of the sides separately and use the INTERPOLATE statement to generate the nodes. A possible command sequence is:

\[
\text{CURVE 0,0 0,1:SIDE1}
\]
\[
\text{CURVE 0,1 1,1:SIDE2}
\]
\[
\text{CURVE 1,1 1,0:SIDE3}
\]
\[
\text{CURVE 1,0 0,0:SIDE4}
\]
\[
\text{INTERPOLATE SIDE1 NODES=3,LINEAR}
\]
\[
\text{INTERPOLATE SIDE2 NODES=5,LINEAR}
\]
\[
\text{INTERPOLATE SIDE3 NODES=4,LINEAR}
\]
\[
\text{INTERPOLATE SIDE4 NODES=6,LINEAR}
\]
\[
\text{REGION SIDE1 SIDE2 SIDE3 SIDE4:SQUARE}
\]
\[
\text{LET 0.3,0.5 0.7,0.5=INTERNAL NODES FOR SQUARE}
\]
\[
\text{MESH SQUARE WITH THREE NODED TRIANGLES}
\]
\[
\text{PLOT MESH}
\]
In this case a plot like Figure 4 would be generated.

Figure 4

To generate a mesh with a different element type, he could type:

MESH SQUARE WITH SIX NODED TRIANGLES

or another available element type. In this case he could produce a plot like Figure 5.

Figure 5

Typically the user will want to specify boundary conditions, regional properties, and global constants. Consider the equation,
\[ \nabla \psi = K \psi + C \]

For instance, the user might type:

```
SET PSI=0.4 ON SIDE1, PSI=0.0, ON SIDE2
SET K=13.1, C=0.0
EXECUTE " SQUARE SOLUTION 
PLOT CONTOUR MAP OF SOLUTION, LEVELS=(0,0.1,0.2,0.3)
```

This would solve the finite element equation and plot the results in a contour map like Figure 6.
Part 2
The FEHPOL Commands

The ASSIGN command allows the assignment and reassignment of the following FEHPOL logical units:

(1) SOURCE
(2) LISTING
(3) DISPLAY
(4) KEEPFILE
(5) REPLY

The format of the command is:

ASSIGN integer TO fehpolunit

where the integer is the FORTRAN logical unit number of the various devices and fehpolunit is one of the above. SOURCE is the stream from which commands are issued, LISTING is the stream to which FEHPOL records are listed on input, DISPLAY is the channel to which WRITE statements output information, KEEPFILE is the direct access file to which the KEEP statement performs I/O, and REPLY is the stream on which error messages and system responses come.

The DELETE command is used to delete defined types from the symbol pool. It has the following format:

DELETE nameofsymbol

It is most commonly used interactively when a user wishes to get rid of a misdefined CURVE or REGION.
The solution program running under the FEHPOL System is executed by a command of the following form:

EXECUTE "titleofsolutionprogram"

When the solution program finishes and makes the return to system supervisor call processing resumes with the next FEHPOL command.

The GENERATE command generates a set of internal nodes to be with a previously defined REGION for finite element mesh generation. Its format is:

GENERATE INTERNAL NODES FOR nameofregion

where the name is of the previously defined REGION.

The INTERPOLATE command takes the set of points representing a CURVE and replaces them with a larger set of points defining it according to one of four interpolation formulae. Its format is:

INTERPOLATE nameofcurve NODES=integer,type

where the name of the CURVE to be interpolated is the first argument, the number of points the user wishes to represent the curve is the integer, and type is one of the following:

1. LINEAR
2. REGULAR
3. SMOOTH
4. CIRCULAR

LINEAR interpolation generates the new points along the straight line segments joining the original points in the CURVE. The int-
Interpolation type, REGULAR, adds new points between the old ones in accordance with a smooth interpolation formula. SMOOTH interpolates points along the same curve but does not retain the original points and spaces the points according to their original density on the curve.

CIRCULAR interpolation is a special interpolation type which is only allowed on CURVES consisting of three points. It defines new points along the circular arc connecting the three points.

The KEEP statement stores the exact configuration of the model at the time of issue. It works in conjunction with the UNKEEP command to save and restore the state of the model. The command is simply:

```
KEEP integer
```

where integer is an index to refer to in the UNKEEP command.

The LET command allows direct user specification of internal nodes for a region. Any reference that is allowed in a REGION or CURVE command is allowed. Its format is:

```
LET listofreferences=INTERNAL NODES FOR nameofregion
```

For example, the command:

```
LET 44,47.2 TTT(6)=INTERNAL NODES FOR HARBOUR6
```

associates the points with the REGION HARBOUR6.

The MESH command is used to generate a finite element mesh on a given region. The mesh is formed by connecting the points defining the region and the internal points associated with it by
the GENERATE or LET commands into finite elements. In addition, the node numbering scheme is automatically optimized for minimum bandwidth in the resulting matrix equations typical of FEMs. An optional feature allows the smoothing of the mesh once it has been connected. The format of the command is:

```
MESH nameofregion WITH elementype
or MESH nameofregion HOLES=(listofregions) WITH elementype
or either of the above followed by
,SMOOTH
```

where the allowable element types (at present) are:

1. SIX NODED TRIANGLES
2. THREE NODED TRIANGLES

The optional SMOOTH parameter repositions internal nodes generated by the GENERATE statement only.

The NODES, CURVE, and REGION commands are all similar. The representations are essentially the same but the meaning is different. The commands are used to associate a list of points and a name. References may be made to previously defined REGIONS, CURVES, and NODES in several ways. The format of these commands is:

```
NODES listofreferences :nameofnodes
or CURVE listofreferences :nameofcurve
or REGION listofreferences :nameofregion
```

For instance, one could issue the command:
CURVE 200,2.5 233,4 343,5 0.577,3:SEAPROB

which would associate the name SEAPROB with the given points and the type 'curve'. In later definitions it could be referenced symbolically like:

REGION 200,2.2 SEAPROB 2655,8:H7244

The effect of this command is as if the points representing the curve SEAPROB had replaced the reference to it. When parameters are set on REGIONs, every element within the region has the parameter set. On CURVEs, every node on the curve (and between the nodes on the curve if side noded elements are used) has the parameter set.

The names of NODES are prefixed in the symbol table by the character colon (:). In like manner, the names of REGIONs and CURVEs are preceded by ampersand and at (& and @).

The PLOT command looks like:

PLOT plottingtype

where plottingtype may be any of the following:

(1) MESH
(2) AND LABEL MESH
(3) NODAL COORDINATES
(4) nameofcurve
(5) nameofregion
(6) nameofnodes
(7) CONTOUR MAP OF nameofregion,LEVELS=(listofnumbers)
The QUIT command reinitializes the FEHPOL System for the next problem. It clears the pools and resets all pointers. It is of the form:

QUIT

The SET command is used to define and pass data to the solution program. Finite element data may usually be classified into three categories: global parameters (i.e., constants, flags) nodal parameters, and elemental parameters. To define a global parameter the format is:

SET name = listofnumbers

The solution program, by referring to the name in a supervisor call can retrieve the value of the numbers. To set nodal parameter the format is:

SET name = number ON nameofcurveornodes
or
SET name = number ON #integer

In the first case, the parameter is set to the number on every point in the CURVE, REGION, or NODE. In the second case it is set on the node whose optimized node number is integer.

To set an elemental parameter the format is like the first of the above, except the name is of a REGION. Every element in the region will have the parameter set to the number.

The TRACE command provides a feature for monitoring the interpretation process. Options are TRACE ON or TRACE OFF. The default setting is TRACE OFF.
The UNKEEP command is used in conjunction with the KEEP command to restore a previous system configuration. Its format is:

```
UNKEEP integer
```

where the integer is an index referencing which KEEP to restore from.

The WRITE command is of the form:

```
WRITE writetype
```

where writetype is any of the following:

1. `nameofcurve`
2. `nameofregion`
3. `nameofnodes`
4. `CONNECTIVITY`
5. `NODES`
6. `nameofparameter`
7. `MESH SPECS`
8. `BANDWIDTH`

This causes the display of the appropriate data on the DISPLAY output channel.
Part 3

FEHPOL System Aborts and Corrective Action

A FEHPOL Abort message and subsequent reinitialization of the system occurs in event of certain conditions. Most often it simply follows the discovery of a condition which should never exist, implying a corruption of the system or the system’s data structure. The reasons for it must be (1) a FEHPOL System bug, (2) a local compiler bug, (3) hardware errors, or (4) operating system bug. The existence of numerous checks for illegal conditions adds to the robustness of the system.

When certain size limitations are reached an Abort message is displayed. The number of such restrictions is small because most data is held in the dynamic symbol and parameter pool with a single limit on the size of the entire pool.

The following paragraphs detail the FEHPOL Aborts and suggest corrective action.

Abort #1-#28: These are reserved for the use of the solution program via the FPSTOP supervisor call.

Abort #29: This occurs when the last symbol or parameter put in the pool has caused it to overflow.

Corrective Action: By programming in FEHPOL more efficiently it is often easy to circumvent pool size restrictions. For instance, by not defining any unnecessary CURVES, REGIONS, NODES, or parameters, one saves pool space. The pool requirements are different from machine to machine since the entry size depends on the local word length. The best guideline is to look at the table following any WRITE statement referring to a symbol. The table gives the current length and amount of space left in the pool.
The following is the procedure for changing the pool sizes.

(1) Find the current total pool size by looking in the system's source file for it initialization:

\[ PT = \text{number} \]

(2) Replace this statement with:

\[ PT = \text{new pool size} \]

(3) In the same segment the symbol pool size is initialized by the statement:

\[ PS = \text{number} \]

The parameter pool is of length PT-PS and the symbol pool is of length PS. Therefore, by replacing this statement the proportion of total pool space devoted to the symbol and parameter pools is altered.

(4) Finally, a multiple edit must be done which replaces every occurrence of 'PL(old pool size' by 'PL(new pool size'.

Abort #30: A parameter pool corruption has occurred.

Corrective Action: If the parameter pool was nearly full before the instruction causing the error, it is likely to have become overfull and the resulting corruption diagnosed before the overflow. In this case, do the same as for #29, otherwise consult the author.
Abort #31: A symbol pool corruption has occurred.
Corrective Action: Same as for #30.

Abort #32: Reserved.

Abort #33: An illegal interpolation code was given to the routine which performs interpolation.
Corrective Action: Consult the author. This can only occur if there has been a corruption of the system's source, codebody, or a hardware malfunction. It is simply a robustness check, which may prove useful when modifications to the system are made.

Abort #34: The code representing the finite element type is an illegal value.
Corrective Action: Same as for #33.

Abort #35: A negative number of internal nodes has been calculated.
Corrective Action: Same as for #33.

Abort #36: Too many equally attractive connections are available.
Corrective Action: This is cause when either more than ten node points lie on the circle of most suitable node to connect to any element side, or equivalently, when two finite element nodes are so close that computationally all others seem equally attractive to connect to. The remedy is to reposition the nodes, as they would form an incredibly poor mesh, anyway.
Abort #37: Too many finite element nodes for the current system size.

Corrective Action: The system must be regenerated with a larger mesh area. This can be done, as follows:

1. Find the statement in the routine FPNCD:

   REAL X(number), Y(number)
   and INTEGER CY(6, another number).

2. Replace all occurrences of '(number)' by '(maximum number of nodes wished)' and every occurrence of '(another number' with '(max number of elements wished). Note the absence of the right parenthesis is important!

Abort #38: Renumbering table corruption.
Corrective Action: Same as for #37

Abort #39: Too many corner nodes.
Corrective Action: Same as for #37

Abort #40-#45: Reserved for system development.

Abort #46: Not enough pool space for INTERPOLATE.
Corrective Action: Same as for #29.
APPENDIX C

Adding Solution Programs to FEHPOL
Appendix C.

Adding Solution Programs to FEHPOI

To run under FEHPOI supervision, existing finite element programs (EFEPs) must be modified to accept and return data to the FEHPOI System. In addition, certain changes must be made to allow entry from and exit to the FEHPOI System's root segment. Certain source statements in the EFEP must be replaced by equivalent FEHPOI supervisor calls to one of the service routines discussed in Section 3.3 of this thesis. After correctly modifying the EFEP source module, it must be compiled, then linked with the FEHPOI System. This appendix describes how to interface an EFEP written in FORTRAN to the system. For EFEPs written in ALGOL or other languages, consult the author.

Step 1. Entry Modification

Precede the first source statement of the main segment of the EFEP by the statement:

```
SUBROUTINE USRPRG
```

Also, make sure no routine or common has a name beginning with the two letters FP. If one does, change it.

Step 2. Exit Modification

Replace each occurrence of the STOP statement by a call to FPSTOP. If the STOP has no argument, use zero.

Example:

<table>
<thead>
<tr>
<th>Original</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=B**2</td>
<td>A=B**2</td>
</tr>
<tr>
<td>10 STOP</td>
<td>10 CALL FPSTOP(0)</td>
</tr>
<tr>
<td>20 IF(A.EQ.B)STOP</td>
<td>20 IF(A.EQ.B)CALL FPSTOP(5)</td>
</tr>
</tbody>
</table>
Step 3., Fetching Mesh Statistics

Find out where in the EFEF the number of nodes, number of elements, and similar information is read. A call to FFNPMS will retrieve from the FEHFOL System the number of elements forming the mesh, the number of corner nodes, the number of nodes total, the number of nodes per element (if constant over the whole mesh), and the semi-bandwidth. The READ statements must be deleted.

Example:

Original

C READ IN NUMBER OF ELEMENTS AND NODES
READ(5,102)NUMEL,NUMNDS
C THIS PROGRAM USES ONLY THREE NODED TRIANGULAR FINITE
C ELEMENTS AND SIMPLY CONNECTED REGIONS
C CALCULATE NUMBER OF BOUNDARY NODES
NUMBN=(NUMNDS-1)*2-NUMEL

Modified

C READ IN NUMBER OF ELEMENTS AND NODES
C READ(5,102)NUMEL,NUMNDS
C *** FEHFOL INSERTIONS
CALL FFNPMS(NUMEL,NDUMMY,NUMNDS,LTYPE,NDUMMY)
IF(LTYPE.NE.3)CALL FFSTOP(11)
C ***
C THIS PROGRAM USES ONLY THREE NODED TRIANGULAR FINITE
C ELEMENTS AND SIMPLY CONNECTED REGIONS
C CALCULATE THE NUMBER OF BOUNDARY NODES
NUMBN=(NUMNDS-1)*2-NUMEL

The call to FFSTOP is used like a FORTRAN STOP statement - to return to the system and display an ABORT message whenever the user attempts to solve a problem using a mesh containing other than three noded elements.
Step 4. Fetching Connectivity Information

Replace the READ statement(s) which input the mesh connectivity by a call to FFEICY.

Example:

Original:

```
C READ CONNECTIVITY
READ(5,101)((ICON(I,J),J=1,4),I=1,NUMEL)
101 FORMAT(416)
```

Modified:

```
C READ CONNECTIVITY
C READ(5,101)((ICON(I,J),J=1,4),I=1,NUMEL)
C101 FORMAT(416)
C **** FEHPOL INSERTIONS
    DO 10 I=1,NUMEL
      CALL FFEICY(ICON(I,1),I)
C ****
```

Step 5. Fetching Coordinates of Nodes

Replace the READ statements which read the coordinates of the finite element nodes by a call to FPNCDS. Since many EFEPs renumber nodes for bandwidth minimization, they use a renumbering table which relates the node number (in the connectivity table) with an optimized node number. To allow easy modification of the EFEP in this case, yet equal facility for EFEPs which do not use a renumbering table, the call to FPNCDS returns the optimized index as well as the coordinates of the requested node. That is, the supervisor call:

```
CALL FPNCDS(5,M,X,Y)
```

causes X and Y to be returned as the coordinates of the node whose FEHPOL optimized node number is M. The first argument is merely a sequencing index.
Example (EFEP without renumbering table)

Original

C INPUT COORDINATES OF NODES
DO 10 I=1,NUMNODES
10 READ(5,102)J,X(J),Y(J)

Modified

C INPUT COORDINATES OF NODES
DO 10 I=1,NUMNODES
C 10 READ(5,102)J,X(J),Y(J)
C *** FEHPOL INSERTIONS
10 CALL FPNGDS(I,J,X(J),Y(J))
C ***

Example (EFEP with renumbering table)

Original

C INPUT COORDINATES OF NODES
DO 10 I=1,NUMNODES
 10 READ(5,104)J,X(J),Y(J)
CALL RENUMBER(CONNECTIVITY,NUMTAB)
C NOW X(NUMTAB(I)) IS THE X-COORDINATE OF THE NODE WITH OPTIMIZED
C NODE NUMBER I

Modified

C INPUT COORDINATES OF NODES
DO 10 I=1,NUMNODES
C 10 READ(5,104)J,X(J),Y(J)
C CALL RENUMBER(CONNECTIVITY,NUMTAB)
C *** FEHPOL INSERTIONS
10 CALL FPNGDS(I,NUMTAB(I),X(I),Y(I))
C ***

Step 6. Fetching Solution Parameters

All other data which the EFEP reads must be replaced by calls to
FPQERY and FPSPMS which fetch solution parameter statistics and their
values. The arguments required for these calls are discussed in detail in Section 3.3 as well as in comments in the FEHPOL System source. In these calls, data items are given a symbolic name by which they can be referred to in FEHPOL.

Example

Original

```fortran
C READ IN DECAY COEFFICIENT
READ(5,106)DECAY
C READ IN TEMPERATURE ON EACH ELEMENT
READ(5,107)(TINIT(I),I=1,NUME)
```

Modified

```fortran
C **** FEHPOL INSERTIONS
DECAY=FPSPMN(5;DECAY,5,1,IX)
IF(IX,NE.0)DECAY=default value for decay coeff.
DO 10 IX=1,NUME
  TINIT(I)=FPSPMN(11;TEMPERATURE,11,1,IX)
  IF(IX,NE.0)TINIT(I)=default value for temperature
10 CONTINUE
C ****
```

Step 7. Returning data to FEHPOL

Any data may be passed back to the FEHPOL System for plotting or display by replacing the WRITE statements in the EEP by calls to FISPUT. This service routine is described in Section 3.3.

Example

Original

```fortran
DO 10 I=1,NUMNCT
10 WRITE(6,99)U(I),ULX(I),DUDY(I)
```
Modified

C DO 10 I=1,NUMNODES
C WRITE(6,99)U(I),LUDX(I),DUDY(I)
C **** FEHPOL INSERTIONS
    CALL FPSPUT(14,STREAMFUNCTION,14,NUMNODES,U)
    CALL FPSPUT(9HXVELOCITY,9,NUMNODES,DUDX)
    CALL FPSPUT(9HYVELOCITY,9,NUMNODES,DUDY)
C ****

Step 8.. Compile Modified EEPF

Making sure that no input or output statements refer to channels 4, 5, 6, or 7, compile the modified program.

Step 9.. Link with the FEHPOL System

Linking the system without any overlays may be extremely wasteful of space. The following simple overlay structure is easy to implement and uses less memory.

FEHPOL ROOT OVERLAY
SEGMENT. INCLUDES:
FPSTOP, FPEXST, FPSPMS
FPQUERY, FPELCY, FPNCDS
FPKMS, and FPSPUT.

The Rest of the FEH-POL System.

Attach here the modified EEPF with its original overlay structure intact.
Sample Link Control Records for FEHPOL/DSPSTD. (Texas 960 Overlay Link Editor)

NORMAL CVLY,MAXSEG=20
ROOT: MAIN
INCLUDE 15(MAIN,FPEXST,FPSTGP,FPSTOP,FPQUERY,FPILCY,FPESBS,FPPKST,
FPPLM,FPFACD,FPSPS,FPRGNR,FPPLM,FPFACD,FPSPS,FPRGNR)
LIBRARY 9,18 .(COMP/COPY library and FORTRAN library)
SEGMENT 1
INCLUDE 15(FPSSTM,FPPCNR,FPACCD,FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP,
FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP,FPSTGP)
LIBRARY 9,17,18 .(COMP/COPY, plotting, and FORTRAN library)
SEGMENT 1
INCLUDE 16 .(all of DSPSTD)
LIBRARY 18 .(FORTRAN library)

Sample JCL Sequence (Texas 960 BASIC Operating Syster.)

//ASSIGN,B0,MHD,FILE0. (Assign DSPSTD Scratch Files)
//ASSIGN,B1,MHD,FILE1.
//ASSIGN,B2,MHD1,FILE2.
//ASSIGN,B3,MHD1,FILE3.
//ASSIGN,B4,MHD,KEEP. (Assign KEEPfile.)
//ASSIGN,B5,VDU. (Commands from Visual Display Unit)
//ASSIGN,B6,LP. (Reply and display to line printer.)
//ASSIGN,B7,DUMMY. (Ignore listing.)
//EXECUTE,MHD1,FEHPOL. (Run FEHPOL.)
Appendix D

Theory and Working of Solution
Program NAVSTK
Appendix D.

Theory and Working of Solution Program
NAVSTK

The pressure-velocity formulation of the Navier-Stokes Equa-
tions and incompressibility condition are:

\[
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + \frac{1}{\rho} \frac{\partial p}{\partial x} - \mu \nabla^2 u = 0
\]  
(1)

\[
\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + \frac{1}{\rho} \frac{\partial p}{\partial y} - \mu \nabla^2 v = 0
\]  
(2)

\[
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0
\]  
(3)

The program considers the following velocity boundary conditions:

\[ u = u_c \text{ on } S_1 \]

\[ v = v_c \text{ on } S_2 \]

\[ \frac{\partial u}{\partial n} = g_x \text{ on } S_2 \]

\[ \frac{\partial v}{\partial n} = g_y \text{ on } S_2 \]

where \( S = S_1 + S_2 \) is the boundary of the problem region. Using the method of weighted residuals the integral formulation used is:
The program uses simple three noded triangular elements with linear shape functions \( L_i \ i=1,3 \). The approximate element solution is:

\[
\begin{align*}
\mathbf{u} &= L_1 \mathbf{u}^i + L_2 \mathbf{u}^2 + L_3 \mathbf{u}^3, \\
\mathbf{v} &= L_1 \mathbf{v}^i + L_2 \mathbf{v}^2 + L_3 \mathbf{v}^3, \\
\mathbf{p} &= \frac{1}{V} \mathbf{p} = L_1 \mathbf{p}^i + L_2 \mathbf{p}^2 + L_3 \mathbf{p}^3
\end{align*}
\]

where \( \mathbf{u}^n, \mathbf{v}^n, \) and \( \mathbf{p}^n \) are the nodal values of velocity and pressure.

This can be written more generally:

\[
\begin{align*}
\mathbf{u} &= \mathbf{L}^T \mathbf{u}^n, \\
\mathbf{v} &= \mathbf{L}^T \mathbf{v}^n, \\
\mathbf{p} &= \mathbf{L}^T \mathbf{p}^n
\end{align*}
\]

where \( ^T \) indicates matrix transpose. Substituting into equations (8) and (9) and integrating over the element, we get:

\[
\begin{align*}
\mathbf{m} \frac{\partial \mathbf{u}^n}{\partial t} + \mathbf{c} (\mathbf{u}^n) \mathbf{u}^n + \mathbf{h} \mathbf{p}^n + \mathbf{k} \mathbf{u}^n &= \mathbf{b}, \\
\mathbf{m} \frac{\partial \mathbf{v}^n}{\partial t} + \mathbf{c} (\mathbf{v}^n) \mathbf{v}^n + \mathbf{h} \mathbf{p}^n + \mathbf{k} \mathbf{v}^n &= \mathbf{b}
\end{align*}
\]
where

\[ m = \int_A \int x \, dA \]
\[ k^* = \oint_A \frac{\partial \mathbf{f}}{\partial x} \cdot \mathbf{dA} \]
\[ \mathbf{c}(x) = \oint_A \int x \, dA \]
\[ \mathbf{a}(y) = \oint_A \int y \, dA \]
\[ h_x = \oint_A \frac{\partial \mathbf{f}}{\partial y} \cdot \mathbf{dA} \]
\[ h_y = \oint_A \int y \, dA \]
\[ \mathbf{b}_x = \sum_{S_2} \mathbf{g}_x \cdot \mathbf{dS} \]
\[ \mathbf{b}_y = \sum_{S_2} \mathbf{g}_y \cdot \mathbf{dS} \]

When the contributions from each element are summed into a global matrix for the whole region, we have the equations:

\[ M \frac{\partial u}{\partial t} + A(u) u + H_x \mathbf{p} + M K_u = \mathbf{B}_x \]  (14)

and

\[ M \frac{\partial v}{\partial t} + A(v) v + H_y \mathbf{p} + M K_v = \mathbf{B}_y \]  (15)

In like manner the global continuity equations are derived from (10) and (11):

\[ H_x \mathbf{u} = \mathbf{B}_m \]
\[ H_y \mathbf{v} = \mathbf{B}_m \]  (16)

where

\[ \mathbf{B}_m = \sum b_m = - \sum \left[ \oint_{S_2} \int_{\mathbf{x}} \alpha \, dS \right] v \]  (17)

and \( \alpha \) is the direction cosine of the normal to \( S_2 \). The resulting
The set of equations is:

\[
\begin{bmatrix}
M & 0 & 0 & 0 \\
0 & M & 0 & 0 \\
0 & 0 & 0 & M \\
H_x & H_y & C & 0
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta v \\
\Delta w \\
\Delta \phi
\end{bmatrix}
+ \begin{bmatrix}
A(u,v) + \tau K & 0 & 0 & 0 \\
0 & A(u,v) + \tau K & 0 & 0 \\
0 & 0 & A(u,v) + \tau K & 0 \\
H_x & H_y & C & 0
\end{bmatrix}
\begin{bmatrix}
u \\
\psi \\
\chi \\
\phi
\end{bmatrix}
= \begin{bmatrix}
B_x \\
B_y \\
B_z \\
B_w
\end{bmatrix} \quad (18)
\]

Approximating the time dependence is done as shown:

\[
\begin{bmatrix}
M & 0 & 0 & 0 \\
0 & M & 0 & 0 \\
0 & 0 & 0 & M \\
H_x & H_y & C & 0
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta v \\
\Delta w \\
\Delta \phi
\end{bmatrix}
= \Delta t
\begin{bmatrix}
A(u,v) + \tau K & 0 & 0 & 0 \\
0 & A(u,v) + \tau K & 0 & 0 \\
0 & 0 & A(u,v) + \tau K & 0 \\
H_x & H_y & C & 0
\end{bmatrix}
\begin{bmatrix}
u \\
\psi \\
\chi \\
\phi
\end{bmatrix}
= \begin{bmatrix}
B_x \\
B_y \\
B_z \\
B_w
\end{bmatrix} \quad (19)
\]

This equation is solved by the program in a modified form, the rows being ordered so the set of equations is of the form:

\[
\begin{bmatrix}
C \\
\sim \\
\sim \\
\sim
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta v \\
\Delta w \\
\Delta \phi
\end{bmatrix}
= \begin{bmatrix}
B
\end{bmatrix} \quad (20)
\]

The matrix C is not positive definite and so either some row-column manipulations, or a solution technique other than Gaussian elimination must be used to solve it.

The program is unfortunately restricted (at present) to a class of problems where the node numbering scheme is carried out manually to circumvent the solution restriction. This caused considerable difficulty in using the program, since either FEHPOL had to be able to order the nodes for NAVSTK's requirements, or NAVSTK needed to be modified to accept a more general numbering scheme.
The method actually used was to generate the problem region description in the required order using FEHPOL commands, and suppressing the optimum bandwidth node numbering scheme. This was not too difficult for the author, knowing the internal workings of FEHPOL, however, it seems clear that the facility should be provided in FEHPOL to specify the numbering system directly.

After the development of NAVSTK, there has been some disagreement as to the legitimacy of the approximation technique used to represent time dependence in the equations. Since then, the programs author has suggested a new approach.
Appendix E

Expanded Introduction and Discussion of Previous Finite Element Systems
Appendix E

Expanded Introduction and Discussion of Previous Finite Element Systems

In today's increasingly technological world, engineers and scientists responsible for the design, modification, and protection of our environment have both a greater obligation and a greater opportunity than ever before to find solutions to many of the problems that have accompanied industrial expansion and modernization. Many of today's problems - overcrowding of our cities, contamination of our waterways, depletion of our natural resources, to name a few - would have been minimized had sound planning been exercised and accurate predictive techniques been available. Responsible, multi-disciplinary scientific investigation can help to provide the reliable and effective guidance which the world community needs to combat its ills, and to plan an organized, integrated environment. An important aspect of this investigation is the development of trustworthy prediction methods which require neither exorbitant capital expenditure to use nor prohibitive developmental costs, yet which may be applied to a wide range of problems. Since many problems lend themselves to mathematical formulation and computer analysis, the construction of software tools has become a significant part of the engineer's and social scientists' approach to them. By making efficient use of these tools, improvements in the quality of our environment and rationality of our society can be initiated.
The finite element method is a mathematical technique for solving the differential equations governing many aspects of our environment which is ideally suited to computer implementation. Particularly for mechanical and civil engineering problems, the method has become established as a powerful modelling tool. Each year sees the application of the method to a new type of problem, and the development of more efficient computational techniques to increase the cost-effectiveness of the analyses.

The mathematical approach begins by re-expressing the differential equations in an integral form. This may take the form of a variational statement which is equivalent to the differential equation, a weighted integral which minimizes the error (in an average sense) produced by assuming an inexact solution, or a simple application of the principles of calculus and topology on the original equations which yields a suitable integral form. The most commonly used techniques fall in the first and second category, namely the Principle of Minimum Potential Energy, and the Galerkin Method (one of the methods of weighted residuals).

Once an integral formulation has been achieved, an effort is made to manipulate it into the form which is suitable for specifying the boundary conditions and constraints on the problem. If the equations are linear, the projection of the solution on a finite function space can be substituted in the equations and the resultant integrals calculated, which results in a series of linear algebraic equations. By combining the equations for each element and solving, the approxi-
mate solution(s) on each element can be calculated.

Before the advent of the computer age, a method which required such a large number of calculations was of little practical benefit. Today, however, the speed and efficiency with which computers handle vast quantities of data and perform calculations has made problems such as the solution of simultaneous algebraic equations a relatively simple task. Consequently, the method may be applied to very complicated problems.

It is the primary attraction of the finite element method that it forms a theoretical and computational basis upon which the analysis of a wide range of problems may be reduced to a common form. In addition, the method allows a distinct demarcation between the facets of the analysis which are common to every applicable problem and the parts which describe the specific nature of the problem under consideration. In its computer implementation, this is manifest in extreme modularity in the configuration of finite element programs. This modularity, more than anything else, has contributed to the development of finite element computer systems, and consequently to a generalized approach to some of the problems of nature and civilization.

Although the basic steps involved in the finite element analysis are similar, considerable diversity exists in the computer organization of the method. Programs range from special purpose, poorly documented codes developed by university research students, to large comprehensive finite element systems used by engineering
consultants and industry. Characteristically, university research codes perform at least the following six functions for a single set of integral or differential equations which describe the problem being analysed.

1. Read data describing the problem's location, orientation, and conditions in terms of a manually superimposed finite element mesh,
2. Set up the calculation of the approximate solution on each element,
3. Combine them all into a large, usually banded, system of equations,
4. Apply conditions by operating on the system of equations,
5. Solve the system of equations using one of several techniques depending on problem size and availability of store. (If the problem involves time dependence or nonlinearity, iteration of some type may be necessary.)
6. Print or plot solution values over the problem region.

The beginnings of the larger finite element systems often emerge as university researchers amalgamate and build upon the work of their predecessors. Although such systems begin as poorly evolved and underdeveloped tools, industrial application and feedback together with sensible and creative design management tend to improve systems' effectiveness and their general applicability.
Without these factors the systems fade into obscurity, or, at most, research use at the institution of origin.

The development of the PAFEC System provides a classic example of the evolution of a series of research codes into a major finite element computer system. The PAFEC System, which can be used to solve a wide variety of mechanical structure problems, was originally developed within the Department of Mechanical Engineering at the University of Nottingham. Initially developed by, and for the research use of, faculty and postgraduate research students, the program was aided by support from Ministry contracts and Science and Research Council Studentships and contracts. Evolving in an academic environment, much of the early work was in the development of new and improved finite elements. The system was composed of a series of commonly used subroutines and parts of various research codes. After considerable growth, the need was recognized to concentrate efforts on producing an efficient, effective design tool which could also be used in industrial and commercial applications. In 1971, the convoluted software architecture of the system was replaced by a more fully integrated and organized file-based structure. The major part of the code was rewritten to conform to the new, modular design. With this change, the system began to become more commercially attractive and some industrial use began. By 1976, a limited company was formed to continue development and market the services of the system.

Because of the orientation of the original research, PAFEC
has a wide variety of finite elements available with which to idealize the structures undergoing analysis. These stretch from simple beam elements to the complex, anisotropic, isoparametric, twenty-noded brick elements, and include many specialized elements.

The system also contains a powerful mesh generation facility known as PAFBLOCKS. The facility uses an approach similar to the method of isoparametric coordinates to locate nodes and elements in regularly shaped zones, each of which may be referred to as a PAFBLOCK. This is a considerable convenience and economy for the PAFEC user, since the data representing the topology of a finite element region and individual node locations is usually extensive and interrelated.

An extremely attractive feature of PAFEC is the recent addition of a simple, language-like data input scheme. This scheme allows the user to submit a concise, straightforward problem description, which is compact, readable, and not subject to formatting errors. The following example shows the data requirements for a simple problem of heat flow in a rectangular region. (Authors Note: Although PAFEC is designed primarily for structural calculations involving stress and vibration analysis, it has the capabilities of handling transient and steady flow problems, as a subsidiary feature. This example was used because it provides an excellent comparison with Example 10, the FEHFCL setup of a nearly identical problem.)
TITLE
"POTENTIAL FLOW IN A RECTANGULAR REGION"

NODES   // X,Y
        0,0
        0,4
       24,4
       24,0
        0,2
       24,2

PAFBLOCKS   // ELEMENT_TYPE= 39110 // N1 TOPOLOGY
            1   1,2,3,4

MESH      // REFERENCE= 1       // SPACING.LIST
            1,9

TEMPERATURE   // TEMPERATURE LIST
               100     1,2,5
               0       3,4,6

IN.DRAW    // TYPE.NUMBER= 1    // INFORMATION.NUMBER= 2

OUT.DRAW   // PLOT_TYPE= 36

PAFEC Commands for Potential Flow Problem (see Example 10)
This example clearly shows how the PAFEC System has sought to eliminate time consuming, redundant, and error prone data input by using a language-like input scheme. It also shows some of the data generation capabilities of PAFEC, for instance, the generation of a mesh on a rectangular PAFBLOCK. Although the exact meaning of each command may be unclear to the reader, the general ease of format and the meaningful keywords can be seen to be a useful aid to the PAFEC user. Example 11 shows a similar problem described using FEHPOL. A close correlation between the various commands can be noticed. Although the techniques used to generate the data describing the idealization are different, the segmentation of the problem into logical parts, and the separation of the data generation phase from the solution phase are quite similar.

PAFEC separates the two by dividing the analysis into two parts. The first consists of describing the problem with user prepared data, generating more comprehensive data describing the idealization, and allowing user examination of the data generated via plots or printout. A second run accesses the generated data, (if it was what the user wanted), and solves the problem according to its instructions. While FEHPOL is designed to be more interactive than this, the basic principle is the same.

At this point it may seem as if the development of data generation languages for finite element analysis is the only logical evolutionary trend for any finite element system. This is
certainly not the case. There are several other trends, which each have their particular advantages and features. For the most part, the designed-in structure of the system is dependent on the size and scope of the problems to be solved. To provide historical insight into the development of finite element systems, it is worth discussing a few systems which are quite different to PAFEC. It is not meant to present a forum for comparison of these systems, but to mention certain features of each which show the range of finite element computer system development.

NASTRAN is a quite sophisticated finite element system which was initially developed for the NASA Space Program. Since its origin, in 1966, it has become well known and established in the aerospace and automotive industries and is regarded as a reliable simulation tool. The system is designed to perform static, dynamic, and thermal analysis of complex structures, and has several particularly attractive features. Among them are a procedure-like language for incorporating new analysis schemes, a facility for handling cyclic and reflective geometry in problem descriptions, and a facility for substructuring analyses. The developers of NASTRAN devoted considerable attention to developing sparse matrix techniques, further enhancing NASTRAN's large problem capability.

A major reason for the success of NASTRAN in industrial applications is the conception and design of the organizational and administrative aspect of the finite element system. Instead
of being enacted as an afterthought, appended to an already complex conglomeration of finite element programs, NASTRAN's "Executive System" was considered of primary importance in the system development period. In fact, the development of certain finite element aspects was left until later. The Executive System's duties include initiating and controlling the sequence of program operations, monitoring primary storage, allocating backing storage as required, and administering a comprehensive restart facility. By having its framework designed from the beginning, NASTRAN lent itself to easy modification and extension.

The most important feature of the Executive System is the versatility achieved by the use of DMAP (Direct Matrix Abstraction Programming). A sequence of DMAP instructions may be compiled by the Executive System prior to NASTRAN execution. This is a similar strategy to that used in some operating systems, where job control commands are precompiled to be used later. The DMAP sequence segments the analysis into a series of separate operations, each of which processes information stored in general format data blocks and produces additional blocks to be used by subsequent operations.

The analogy between DMAP and some large scale operating systems job control languages does not stop there. Like many operating systems (Texas DX950, IBM360/OS,...), NASTRAN supports a "procedure library" of already existing DMAP sequences, each tailored to a particular kind of analysis. In this way, the capabili-
ties of the system are kept general without detracting from the system's convenience to the user - at least for the commonly used analyses for which DMAP sequences are already available. These can be used quite flexibly, since NASTRAN allows modifications to be made to existing sequences before execution (similar to the way parameter substitution takes place in many job control procedures). This organizational pattern has proven to be highly efficient for certain applications, and consequently is a good choice for a large finite element system like NASTRAN.

An interesting facility of NASTRAN is the utility which takes advantage of cyclic and reflective symmetry in its finite element calculations. The analysis is formulated so as to avoid the formation of the complete system of equations representing the configuration of the structure. The transformations and back-transformations which characterize this time and storage saving reduction in complexity are handled automatically by the system. In addition, they may be superimposed in the case of a combination of symmetries. This feature is very useful in types of analysis where problems have a high degree of symmetry, as in aerospace applications.

General substructuring capabilities can be achieved by successive modification of existing DMAP sequences using the ALTER facility. Using this facility, the analysis of each substructure can be performed to find its internal relations, then the results are combined to model the whole structure. A final step solves for the internal configuration of each substructure. Using this
Another NASTRAN feature is the facility which initiates and performs restarts. Most finite element systems have some form of restart facility, since calculations using the finite element method can consume considerable time, and the possibility of data errors, machine failure, and software bugs increases with total run time. Besides this, in many analyses it is convenient to restart the analysis from a certain point after having examined the results of an initial run. Particularly in design problems, where the problem description is being varied at the discretion of the designer, the ability to branch off from a given point in a previous analysis in order to obtain optimal solution characteristics is a very economical utility indeed. NASTRAN allows the user to request that all input data and major data blocks generated during a NASTRAN run be saved and catalogued on a backing storage medium. The information can be recalled and modified on a subsequent run under control of the user.

Before going on to a brief discussion of FEHPOL itself, it is important to mention one more finite element system which presents an approach different from that of either of the previously discussed systems.

GENESYS is a computer system for manipulating programs. It is designed to allow the inclusion of existing FORTRAN programs.
with a minimum of modification, and to administer comprehensive
management services related to the organization of finite element
programs and subsystems. Its features include the ability to
compile programs written in a high level extension of FORTRAN IV
called GENTRAN, an efficient virtual store is enhanced by the
system's ability to store arrays in a compacted tree structure, a
well organized documentation scheme, and a large number of existing
analysis subsystems which are fully tested and relatively reliable.

The GENESYS System was developed for the Ministry of Public
Building and Works for use in the construction industry. It was ori-
originally maintained by the GENESYS Centre, but more recently by a com-
pany known as GENESYS Ltd., which distributes documentation and sup-
ports continuing system development on several machines.

All the programs that GENESYS controls and manipulates are
written in GENTRAN. In practice, the precompiler part of GENESYS
operates on the GENTRAN source statements and produces a FORTRAN
program which is compatible with the local computer. GENTRAN is,
however, more than a high-level programming language. This is be-
cause many utilities are included in the final program which were
not coded by the GENTRAN user. They are generated automatically
by the system, and provide the monitoring and other system func-
tions during a GENESYS subsystem run. Because the language is
similar to FORTRAN, engineers find it easy to learn. For many
analyses, however, it is unnecessary for the engineer to program
in GENTRAN at all, since complete subsystems are already available.
As was mentioned briefly before, GENTRAN allows the programmer to define and reference sparse arrays. These use tree structured storage. This means that array elements do not exist until used, and that the storage space for an array need only include those elements which have been allocated. Since most finite element problems involve the inversion of at least one sparse matrix, this is a substantial storage economy.

The most impressive feature of GENESIS is its virtual store. By virtual storage we mean that programs can use what apparently is more storage than is available on the local machine without overflowing the memory limits. Swapping of data back and forth from computer to backing storage is accomplished completely automatically by the system itself. The programmer is unaware of any swapping except for a degradation in speed of execution. One of the beauties of this system is that a simple, relatively crude program will operate on the system with no concern over the availability of storage. Also, the development cost compares favorably with that of a program with sophisticated storage schemes written in. Coupled with the economical array packing techniques, this means that the size of the problems that can be solved using GENESIS is limited only by the availability of time and backing store (or GENESIS user's budget).

GENESIS has several other facilities similar to those of the previously mentioned systems, and a complete discussion of them all is not intended. The data retrieval technique is noteworthy,
however, because of its similarity with the technique used by the FEHPOL System.

The Gentran READ statements, rather than performing the transfer of information from a peripheral device to the program, access items of data from GENESYS' data tables by referring to the items name and type. This means that the data may be "read" over and over again, and in fact, need never be stored within the program being run. GENESYS builds its tables from data supplied by the user before a GENTRAN READ statement is ever encountered. Clearly this is a good organizational feature for finite element analyses, where most of the data is tabular.

In the past several pages three major finite element systems (primarily of the structural engineering variety) have been discussed. While other systems are available for different classes of problems and to meet specialized requirements, the various approaches to designing the software structure and finite element data handling present a general, broad picture of the current state of the art. This provides a background upon which to build a discussion of a new, special purpose finite element computer system, the FEHPOL System.

The FEHPOL System, rather than being a competitor to the large scale systems, is a powerful research and design tool specializing in interactive consulting use. The systems extensive data generation facilities and software interface for adding finite element programs characterize its approach. The system is associated with
the recent development of finite element techniques for fluid flow and related problems of continuous media. In problems of this nature, the need for a large number of highly sophisticated finite element types is not very great. FEHPOL enables the user to describe a problem quickly, simply, and interactively, and achieve an immediate and satisfactory solution.