PREPR2: A Program to Aid in the Preparation of Input Data for the Farout Hydraulic Transport Code

Jonas T. Holdeman, Jr.

NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE. It
has been reproduced from the best available
copy to permit the broadest possible avail-
ability.

MASTER

OAK RIDGE NATIONAL LABORATORY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
Price: Printed Copy \$5.00; Microfiche \$3.00

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the Energy Research and Development Administration, United States Nuclear Regulatory Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Contract No. W-7405 eng 26

COMPUTER SCIENCES DIVISION

PREPR2: A PROGRAM TO AID IN THE
PREPARATION OF INPUT DATA FOR THE
FAROUT HYDRAULIC TRANSPORT CODE

Jonas T. Holdeman, Jr.

NOTICE

This document contains information of a preliminary nature.
It is subject to revision or correction and therefore does
not represent a final report.

Date Published - May 1977

Work sponsored by the Energy Division and supported by the Office of
Nuclear Regulatory Research of the U.S. Nuclear Regulatory
Commission under contract Transport-Assess Power
Plant Impact - Thermal, 60190501, 8016c-6

MASTER

UNION CARBIDE CORPORATION, NUCLEAR DIVISION
operating the
Oak Ridge Gaseous Diffusion Plant . Oak Ridge National Laboratory
Oak Ridge Y-12 Plant . Paducah Gaseous Diffusion Plant
for the
ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION

TABLE OF CONTENTS

LIST OF FIGURES	iv
LIST OF TABLES	v
ACKNOWLEDGMENTS	vii
ABSTRACT	1
1. INTRODUCTION	3
2. THE DISCRETE ELEMENT METHOD	6
2.1 Data Preparation	10
3. PREPR2 - A GEOMETRIC DATA PREPROCESSOR	11
3.1 Main Program	12
3.2 Subroutine PREPR2	15
3.3 Subroutine INITL	15
3.4 Subroutine SHOREL	15
3.5 Subroutine BNDREL	17
3.6 Subroutine MARKEX	18
3.7 Subroutine CENWID	19
3.8 Subroutine REMSBE	21
3.9 Subroutine BNDFIN	22
3.10 Subroutine DATDEK	22
3.11 Subroutine ICROSS	23
3.12 Subroutine SHRMAP	24
4. INPUT DATA FOR PREPR2	25
5. RESULTS OF AN APPLICATION OF PREPR2: ATLANTIC GENERATING STATION	29
REFERENCES	35
APPENDIX A. The Computation of the First Moments . . . . . of a Closed Curve with Particular Application to a Polygon	39
APPENDIX B. Program Listing of PREPR2	45
APPENDIX C. Identification of Some Variables Used in . . . . . the Program PREPR2	69
APPENDIX D. Partial Output Listing From PREPR2 Run, ... Atlantic Generating Station	75

BLANK PAGE

LIST OF FIGURES

1. Internal Element	6
2. Boundary Element	8
3. Functional Flow of PREPR2, Task Oriented	12
4. Line Segment Intersections	19
5. Map of Shoreline Near Proposed Atlantic Generating Station	29
6. Map of Shoreline Near Proposed Atlantic Generating Station With Grid Structure Superimposed	30
7. Reconstruction of Shoreline Geometry by the Program GRAFAR from a Data Set Produced by PREPR2	33

LIST OF TABLES

1. Input Data to PREPR2 for the Atlantic Generating Station Site	26

ACKNOWLEDGMENTS

This report describes one subtask of a larger project involving a "unified transport approach" (UTA) to the modeling of the transport and dispersion of intensive quantities such as heat or chemical or radiological pollutants dissolved, entrained, or supported by the movement of natural waters in rivers, estuaries, or seashores. The UTA project is supported by the Office of Nuclear Regulatory Research of the U.S. Nuclear Regulatory Commission and is managed within the Environmental Impact section of the Energy Division at Oak Ridge National Laboratory by T. H. Row.

The author would like to thank Patricia A. Olson, who wrote the program PREPAR, the predecessor and namesake of PREPR2, for many valuable discussions. Experience gained with PREPAR was invaluable in the design of PREPR2. Thanks also are due to Donald R. Gray and Ed C. Long for discussions about user requirements, to J. M. Barton and Harry Diamant for helpful discussions about the FAROUT program, and to S. K. Iskander for discussions about mesh generation for finite element programs. The help of Benjamin Thomas and Jean Lucas Bledsoe providing graphical support with the program GRAFAR is gratefully acknowledged as is that of A. H. Eraslan, project director and Consultant from the University of Tennessee, and M. R. Patterson, UCC-ND task leader, who provided guidance through many helpful discussions. Finally, we gratefully acknowledge the technical support of the Computer Sciences Division at ORNL.

PREPR2: A PROGRAM TO AID IN THE
PREPARATION OF INPUT DATA FOR THE FAROUT
TWO-DIMENSIONAL DISCRETE ELEMENT CODE

Jonas T. Holdeman, Jr.

ABSTRACT

A computer code PREPR2 was written to aid in the preparation of data sets for the FAROUT¹/FAR2D/FARTMP³/FARCCH/FARCRD computer codes used for the assessment of environmental impacts in power plant sitings. These latter hydraulic transport codes use two types of discrete elements, a rectangular "internal" element and a more complicated "boundary" element which incorporates the irregular shape of the region being simulated. The data set is prepared from a map of the area to be simulated and a description of the discrete element structure to be used. This report describes the computer program, the requirements for its use, and illustrates its use at an actual plant site.

I. INTRODUCTION

The FAROUT* family of hydraulic transport codes is being developed to provide a unified method for modeling the transport and dispersion of intensive quantities such as heat or chemical or radiological pollutants released from power plants and dissolved, entrained, or supported by the movements of natural waters in rivers, estuaries, or seashores. The increasing energy demands of a growing population and the occupation of the more desirable sites dictate consideration of siting in marginal areas where the ecological impact may be quite severe. The assessment of this impact which has been little stressed in engineering design, is now mandated by law so the process of rule of thumb or educated guess must now be replaced with sophisticated predictive models.

The prediction of the course of physical phenomena has long been a subject of interest and the development of quantitative predictions had its basis in the formulation of physical "laws" in mathematical form and the use of the tools of mathematical analysis to reveal useful relations among observable quantities. As powerful as these analytical tools are, they are frequently of limited use when confronted with the full complexity of large physical systems. A successful approach in the analysis of large systems has been to partition the large system in subsystems. The smaller subsystems have fewer degrees of freedom and, if properly chosen, may be adequately described

---

*We will use the generic name FAROUT to indicate any member of the FAR2D/FARTMP³/FARCRD/FARCCH family.

BLANK PAGE

by a few variables or parameters. These subsystems then interact with each other and, if an appropriate partitioning of the original system was effected, the principal interactions may be described by simple rules or mechanisms.

The degree to which the partitioning of a large complex system is carried out forms a natural basis for classifying and comparing computational methods. At the less extreme end the subsystems are allowed a modest amount of internal structure and the freedom of interaction is great enough that the interactions among the various subsystems must be described in some detail. One computational method having these attributes is that of Finite Elements. This method is very elegant and powerful, but its implementation requires a large amount of computer storage. At the other extreme the subsystems may be so small as to have little or no internal structure and interact directly with only a few nearest neighbors. Finite Difference techniques usually involve this limit. Though the number of nodes in a Finite Difference computational scheme may be quite large, the data storage is structured in the same way as the interactions are structured (or maybe the reverse is true) and the computational algorithms usually based on finite difference approximations to differential equations, are very fast and efficient.

There is, of course, a middle ground, and this is where the "Discrete Element"¹ approach used in the farfield code FAROUT¹ is to be found. The bulk of the region simulated is modeled with rectangular elements with very little explicit structure. Direct interactions obtain only

with nearest neighbor elements. The boundary region which contains the land-water interface cannot in general be modeled with the relatively large rectangular elements. The elements which model this interfacial region have much more internal structure and while each boundary element interacts with at most four nearest neighbors, the details of the interaction are considerably more complicated. The details of the internal structure and interactions of these boundary elements must be carefully described to direct the computational algorithm.

In the Finite Element method each element must be carefully described. Mesh generators have been in use in this area for some time to relieve the analyst of some of the tedium of preparing a data set for a large system, as well as to reduce errors inherent in hand preparations. Such mesh generators are used largely for assistance and are usually limited to simple symmetries or to replicating a pattern established by the analyst. These mesh generators are not nearly so elegant and effective when applied to irregularly shaped regions. It is precisely the irregularly shaped boundary where it is necessary to invoke boundary elements when dealing with the "discrete element" computational method. Thus it has not proved feasible to adapt existing mesh generators developed for the Finite Element codes to the "discrete element" problem.

The Discrete Element approach was chosen for the thermal transport modeling implemented in the FAROUT programs. This approach to the spatial variables together with the "method of lines" for treating the time evolution of the system is well suited to the simulation of the transport of other conserved or quasi-conserved quantities and was chosen as the basic computational algorithm for application of a

Unified Transport Approach (UTA) to the simulation of the transport and dispersal in the aquatic environment of thermal, chemical, and radiological discharges from proposed or operating power plants.

This report will describe a geometric data preprocessor, PREPR2, written to generate the geometric data needed for a UTA simulation using the discrete element method. To understand what data is needed and why, we will first review the discrete element method which is used.

2. THE DISCRETE ELEMENT METHOD

In the discussion to follow it is assumed that the region being modeled or simulated is two-dimensional or pseudo-three-dimensional. This is the situation in the implementation of the FAROUT code. The ideas may be extended to one and three dimensions.

The discrete element method attempts to draw together the finer features of the Finite Element and Finite Difference methods into a single computationally efficient algorithm. The Finite Difference method represents subregions as simple, almost structureless elements with a simple implicit interconnection. From this method we draw the simple rectangular element and rectangular pattern of organization. We define a set of indices IX and JY , such that each discrete element has a unique pair of indices and to each pair of indices there is at most one element (some indices may address pseudo-elements called "external elements" which do not interact with any active element). The element with indices IX, JY may directly interact at most with the elements with indices $IX \pm 1, JY$ and $IX, JY \pm 1$, that

is, with its nearest neighbors immediately to the left and right, above and below, and with the external environment.

The simplest elements are rectangular, are characterized by a center or centroid position, a height, and a width, and are called internal elements (Fig. 1). For an internal element with internal neighbors, the centroid, width, and height are not independent. All internal elements in the same column (same index IX) have the same width DXC and their centroids have the same X -coordinate XC . All internal elements in the same row (same index JY) have the same height DYC and their centroids have the same Y -coordinate YC . The absolute position of the centroid is not used but its distance from the edges of the element (which is $DXC/2$ and $DYC/2$) is used in several interpolation formulas and to compute thermal gradients; the widths DXC and DYC are used to compute the surface area of the element (used in some interactions with the external environment) and with the depth variable to compute the element volume, and the vertical element surface through which the fluxes of water, heat energy, and chemical or radiological species pass between elements.

Fig. 1. Internal Element.

The boundary elements (see Fig. 2) are characterized by a pair of indices (IX, JY) , a pair of centroid coordinates (XCB, YCB) and a pair of widths $(DXCB$ and $DYCB)$. These widths are used as before, to compute the surface area and volume of the element but should not be used in the interpolation formulas and thermal gradients. For each face of an element through which interactions with adjacent elements can occur, the length of the active interface and the distance from the centroid to the interface is needed. In the present implementation of FAROUT, the centroid width and centroid-interface distances are not distinguished. For a simple triangular boundary element, the interpolation and gradient applications will be correct but the area and volume computed will be twice too large. While other boundary element shapes can give large errors in the thermal gradient and interpolation formulas as well, the approximation will be adequate for boundary elements which are approximately rectangular.

For discrete element computations involving moving fluids (FLIDE), the velocity at each element is needed in the calculation of transport fluxes. Other quantities needed are the depth as was previously mentioned and the distribution of the temperature, or concentration of chemical or radiological species. The time rate of change of intensive quantities of interest for each element is then developed from an integral form of the appropriate conservation laws.

Interactions with the external environment usually involve fluxes through the horizontal surfaces above and below the element, but a pseudo-volumetric interaction may obtain, as in radioactive decay or chemical reactions.

Centroid coordinates: XCB
 YCB

Interface lengths: $DXBJPH$
 $DXBJMH$
 $DYBIPH$
 $DYBIMH$

(= 0 in Figure)

Interface to centroid distances: $DXCBP$ (not needed here)
 $DXCBM$ ($\approx DXCB/2$)
 $DYCBP$ ($\approx DYCB/2$)
 $DYCBM$ ($\approx DYCB/2$)

Area: $DXCB * DYCB$

Fig. 2 Boundary Element.

2.1 Data Preparation

Having discussed the general data requirements of the discrete element approach, we now consider the ways in which these requirements may be met. The procedure used with FAROUT involves the construction of a grid structure which has small elements in the vicinity of plant intake and exhaust structures or other locations where greater detail or higher spatial resolution is needed. Progressively larger elements are used away from the plant where the thermal plume is larger and less detail is required, but where the quantity of the variable, whose transport and distribution are being simulated, must still be accounted for. With the procedure used in the past, this grid structure was then ruled over a map of the area. The elements of the various types were identified and the geometric quantities necessary to describe or specify the elements measured or calculated. These values were then transcribed to punched cards in the proper format. The process described takes from a week to a month to accomplish and experience has shown that the procedure is prone to errors.

There are valid reasons why more than one element structure and data set may be desirable. Standard error analysis associated with the discretization process is difficult to perform with programs as complex as FAROUT. An alternative would be to do a sensitivity analysis on the site being studied using an alternate array structure to infer the magnitude of the discretization error inherent in the element structure chosen for the final analysis. The magnitude of the task of preparing alternate data sets and the difficulty in eliminating errors have precluded such sensitivity studies in the past.

A better procedure would be to have the topography of the region around the site encoded into a digital map and stored on a data file. This map would ideally include shorelines and bathymetric data and would include the identification of special points of interest. The grid associated with the finite element structure would be designed and may be specified separately. The location and orientation of the grid would be given. Then a digital computer program would merge the two data files, identify elements as to type, compute the quantities necessary to characterize each element and create a data set in the format appropriate for entry into the farfield code. If for any reason it is necessary or desirable to modify the element grid, a new data set can be generated immediately.

The procedure just described has been used as a guide in the design of a geometric data preprocessor code called PREPR2. This data preprocessor has been used both interactively and in batch mode and has proved to be fast and effective. While all of the features originally envisioned have not been included as yet, this computer code has proved to be a valuable aid in the preparation of data sets for FAROUT.

3. PREPR2 - A GEOMETRIC DATA PREPROCESSOR

The general considerations discussed previously were used as a guide in the development of the geometric data preprocessor PREPR2. Additional programming considerations were that the program be written in standard FORTRAN for improved portability, be written in a modular fashion and

structured for easier maintenance, and be simple and reliable to use. Some early design choices were strongly influenced by an earlier code named PREPAR which during development became too complex and cumbersome to expand and maintain.

The overall problem of data preparation was partitioned into several distinct sequentially executed tasks. A module was then coded to execute each task. These modules are supported by additional subprograms which may be shared by the task-oriented modules. At a higher level, the task modules are called in turn by a program element named PREPR2 which is structured as a subroutine. At the highest level, the main program allocates storage space or computer memory to the rest of the program based on the anticipated needs of the problem at hand. A diagram illustrating the task-oriented flow of control is shown in Fig. 3.

3.1 Main Program

The principal function of the main program is to provide and allocate storage for the working modules. The bulk of the working storage is provided by the single array named S with the instruction to the FORTRAN compiler to reserve a specified amount of storage given in the DIMENSION statement. The amount of storage so requested must be made available to the main program itself through the variable NS which is set in the DATA statement following the DIMENSION statement. If the analyst fails to provide enough storage for the problem at hand, execution is terminated with a message giving instructions for modifying the DIMENSION and DATA statements to handle the problem on a subsequent run.

Fig. 3. Functional Flow, Task Oriented.

Several incidental assignments and tasks are performed prior to storage assignment. The logical input and output unit numbers are assigned values; then a title is read which is used to identify the site in the printed and punched output.

The second set of data items read is the pair NXREGN and NYREGN, the number of columns and the number of rows of elements, and three control variables IFPNCH, IFDLST, and IFTRCE. The function of the latter three variables is explained in section 4 of this report describing the input data. The two variables NXREGN and NYREGN indicate the size of the problem and imply the amount of storage needed. The number of storage locations allocated for boundary elements LBND and the number allocated for the segment lists NSEGN are computed with the algorithms,

$$LBND = 3.5 * (NXREGN + NYREGN) ,$$

$$NSEGN = 4 * (NXREGN + NYREGN) .$$

These algorithms have proved adequate; but, if overflow should occur in some unanticipated application, the appropriate algorithm would have to be changed for that application.

The available storage is segmented into blocks of the appropriate length and the starting address of each segment or block and the lengths are passed to the subroutine PREPR2 when it is called. Upon program return from PREPR2, a message is printed indicating the number of words of storage used and not used, for the information of the analyst, should he choose to minimize his use of computer resources on a subsequent run.

3.2 Subroutine PREPR2

The function of Subroutine PREPR2 is the assignment of various tasks by calling the modules or subroutines which execute the tasks. Normally PREPR2 would be the main program, but the division of administrative tasks between the main program and PREPR2 has certain advantages. First, all of the subroutines including PREPR2 may be compiled and stored as a library file used when loading the main program. Second, the arrays used in PREPR2 are variably dimensioned and may be referred to by names with mnemonic value, the same names which are used elsewhere in the program. PREPR2 calls the subroutines INITL, SHOREL, MARKEX, CENWID, REMSBE, BNDFIN, PRINIT, and DATDEK in that order to complete each of the tasks.

3.3 Subroutine INITL

Subroutine INITL is the first subroutine called by PREPR2 and its task is to perform whatever initialization operations are necessary, including the reading of any input data. Its first action is to read the coordinates of the element mesh or edge coordinates of the internal elements. INITL then calls the service subroutine SHRMAP and signals that subroutine to perform the initialization it needs which usually includes reading the map data.

3.4 Subroutine SHOREL

Subroutine SHOREL and its companion subroutine BNDREL perform the task of following shorelines, detecting entries into elements, tracing shorelines through elements while accumulating information for the computation of enclosed areas and moments, detecting exits, identifying edges, measuring

wetted lengths of these edges of the element and completing the area and moment computations. From this description it is easy to see that this is the most complicated and difficult of the tasks assigned by PREPR2.

For each shoreline, if the first point is exterior to the array of elements, then the shoreline segments are examined sequentially until an intersection with the edge of the array of elements is found; then the coordinates of the point of entry and the indices of the element entered are calculated. If the first point on the shoreline is interior to the array, then the shoreline must be closed (i.e., an island or a lake); otherwise, the shoreline information is insufficient to define some of the boundary elements. A closed shoreline is indicated by the condition that the coordinates of the first and last points on the shoreline are identical. If the shoreline is closed, then the initial element will be reentered later, so any processing of this initial element is deferred. The shoreline is simply followed until it leaves the initial element and the side and coordinates of the exit point are found. This gives the point and side of entry of the next element at which normal shoreline tracing is initiated with a call to subroutine BNDREL.

When the shoreline terminates inside of an element being processed in subroutine BNDREL, the terminal point on the shoreline is compared with the initial point on the shoreline upon return to SHOREL and the tracing of the shoreline within the element continues with the first point of the shoreline until the shoreline exits this element which has the initial element. A warning is in order here. If the initial (and terminal) point of a closed shoreline lies on the edge of an element (on one of the grid lines), the closure of the shoreline will not be recognized and the properties of the last boundary element on that shoreline will be in error.

3.5 Subroutine BNDREL

Subroutine BNDREL is given an element, the side of that element which the shoreline enters and the coordinates of the point of entry. BNDREL then traces the shoreline until it exits the element and returns to SHOREL the coordinates of the point of exit, the side of exit, and a flag which indicates whether SHOREL is to continue with that shoreline. Before returning, however, BNDREL completes the calculation of the assigned geometric properties of the element.

Upon entry to BNDREL, the properties of the nominally rectangular element for the specified indices are computed and the value stored in array IELTYP is checked to see if the element is a new boundary element or an old boundary element being reentered. If the former, a new boundary element is created; and if the latter, the reentry information is stored in the linked list, LSEG.

A trace of the shoreline is then begun. The centroids are to be calculated from the area and first moments. These latter quantities are calculated using formulas given in Appendix A. When the shoreline exits the element, the coordinates of the exit point and the edge are found and recorded. Computation of the areas and moments is finished and the edge conditions and water lengths along the edges are evaluated. Finally, multiple entry corrections are made.

Another warning is in order here. BNDREL is supposed to return the side on which the shoreline exits. If the shoreline exits precisely through a corner, the "side" of exit is ambiguous and an error will occur. Should this happen, a grid line or one of the endpoints of the offending line segment should be displaced slightly.

3.6 Subroutine MARKEX

Subroutine MARKEX identifies any external elements which may be present and marks them as external by setting the entry in the array IELTYP corresponding to the external element to the value -1. Initially all elements were assigned a default type of "internal" by the entry of the value 0 in the array IELTYP. As boundary elements were identified, they were marked as boundary by the entry of a positive value in the array IELTYP.

Two principles are used to identify external elements: (1) No internal element is allowed adjacent to an external element, and (2) if there exist (possibly adjacent) external elements, then at least one of them must occur adjacent to a boundary element. Using the latter principle, the list of boundary elements is scanned and the perimeter checked for an edge marked external. When an external edge is found, the element adjacent to this edge (if the indices of the element are valid or in range) must be external (by condition 1) and is so marked. The elements adjacent to the three remaining edges of this external element are checked. Any element not marked as boundary must be type external. It is so marked and the elements adjacent to its three edges are checked. The process continues until no more elements are found to be marked. Then the next boundary element is examined for external edges.

The algorithm just described is best implemented using recursive subroutine calls. Since recursive calls are not permitted in FORTRAN, the algorithm is implemented using loops and a push down stack.

Occasionally the line of boundary elements will be incomplete so that the external and expected internal regions are not separated. This might

occur if a shoreline did not leave the region being simulated because of an error in a datum entry or a failure to provide data extending the shoreline, or in certain cases where a shoreline segment crosses an element corner. In that event all non-boundary elements may be marked as external, usually with a message from MARKEX indicating a stack overflow. The map of the elements produced by PRINIT is useful in diagnosing this situation.

3.7 Subroutine CENWID

Subroutine CENWID determines the widths of each boundary element measured through the element centroid in conformance with that definition given by J. M. Barton.² Clearly this cannot be done until after the centroid positions have been determined by subroutine SHOREL. For regular boundary elements bordering regions of open or specified flow, the computation is trivial so that the centroid widths for those elements are determined when subroutine BNDFIN is called. The centroid widths are also dependent on the shoreline which was traced by SHOREL to determine the centroids. Rather than retrace the whole shoreline in CENWID, Subroutine SHOREL stores information about that part of the shoreline which intersects each element in a linked list, LSEG. Retrieval of this information simplifies the task considerably.

For each boundary element, initial centroid widths of zero are assigned. Then each line segment which is known from information gathered in SHOREL to intersect the element is examined for an intersection with one of the horizontal or vertical line segments drawn through the centroid and lying interior to the segment. If an intersection obtains interior to both line segments as in Fig. 4a, that portion of the line segment through the

centroid lying to the left of the shoreline segment is identified by the sign of a vector cross product and its length calculated. This length is added to the previously calculated centroid width (which will be zero if this is the first interior intersection). The process is repeated for each shoreline segment.

On completion of the examination of the set of shoreline segments, the widths may be found to be greater than the rectangle widths if multiple intersections were detected as in Fig. 4c. The width will be too large by some integer multiple of the rectangle width, and the width modulo-the-rectangle-width is computed by repeated subtraction of the rectangle width. Finally, if the width is zero, as would be the case if one too many subtractions were performed or if no interior intersections were found as in Fig. 4b, the width is set to the rectangle width.

Fig. 4a
Interior
Intersections.

Fig. 4b
Exterior
Intersections.

Fig. 4c
Multiple
Intersections.

3.8 Subroutine REMSBE

The function of subroutine REMSBE is to eliminate small or unnecessary boundary elements by combining two or several into a single boundary element. In the FAROUT program the largest integration step size permitted, and hence the efficiency, is proportional to the areas of the elements and is often dominated by the area of the smallest element. It is therefore advantageous to eliminate the smallest boundary elements by combining them with adjacent boundary elements where possible. An added computational benefit results from the reduction of the number of active elements.

Smallness is a relative concept so the array of elements is scanned to find the smallest internal element. The size of the smallest internal element was probably chosen by the analyst to achieve a desired spatial resolution and would naturally bound the time interval. Boundary elements are candidates for elimination when they are smaller than this smallest internal element and are eliminated when their area is less than $FRAC \times$ (smallest internal area) and one side is or becomes external. When a boundary element is eliminated in this manner, its edges are checked and its area is combined with the adjacent active element with the largest water interface. If this active element happens to be an internal element, it is first changed to a boundary element.

Boundary elements are also eliminated and their area combined with an adjacent boundary element when they are surrounded on three sides by external elements. As described before, if the adjacent element is not a boundary element, a boundary element is created in that place.

Each modification of the boundary element structure may permit additional modifications. Thus the execution of the subroutine is repeated until a pass is made through the subroutine without making any changes to the boundary element structure.

3.9 Subroutine BNDFIN

Those active elements which are boundary elements by virtue of being cut by a shoreline have already been identified and marked in subroutine SHOREL. There remains a set of elements which would be internal but for the fact that they lie on the edge of the grid. The properties of their missing neighbors must be specified externally or must be inferred by extrapolation from the interior of the region being simulated. The exceptional treatment required to compute the flow into these elements makes them of the boundary type.

At the point in the computation when BNDFIN is called, the subroutines SHOREL and MARKEX have been called so any remaining elements around the border of the array of elements with IELTYP = 0 (still marked as internal) must be boundary elements. These elements are now marked as boundary elements and their dimensions are computed and stored and edges marked.

3.10 Subroutine DATDEK

Subroutine DATDEK completes the final task assigned by PREPR2, the generation of a data deck in the format required by FAR2D. The output is under the control of two logical variables LPUNCH and LPRINT which were set before the first task was initiated. A data deck is written on logical unit DATOUT in punched card format when LPUNCH is true, no data errors have been detected, and logical unit DATOUT is distinct from the

print unit OUT. A copy of the output data file is written on unit OUT in printer format when the logical variable LPRINT is true or when an error is found while generating the data deck.

3.11 Subroutine ICROSS

Subroutine ICROSS is a service routine called from subroutines SHOREL/BNDREL and CENWID to find if two directed line segments intersect and, if so, to compute the point of intersection. Some special conventions are used with intersecting colinear segments. The line segments are specified by giving the coordinates of each initial point and the coordinates of the terminal point relative to the first. The points on each line segment are represented by the parametric equations

$$X(S) = X_S + DX_S \cdot S ,$$

$$Y(S) = Y_S + DY_S \cdot S, \quad 0 \leq S \leq 1,$$

and

$$X(T) = X_t + DX_t \cdot T ,$$

$$Y(T) = Y_t + DY_t \cdot T, \quad 0 \leq T \leq 1.$$

When the segments are not parallel as indicated by the relation $DY_S \cdot DX_t \neq DY_t \cdot DX_S$, the values of S and T corresponding to the intersection point of the extended lines are returned. Intersection of the segments is then indicated by the conditions $0 \leq T \leq 1$ and $0 \leq S \leq 1$.

If the segments are parallel and not colinear, this condition is indicated by returning large values of S and T. When the segments are colinear, some special conventions needed by subroutine BNDREL are returned.

3.12 Subroutine SHRMAP

Subroutine SHRMAP is a service subroutine with the calling sequence

CALL SHRMAP (NSHOR, NP, X, Y, NC).

The detailed implementation of SHRMAP may vary with the application of PREPR2 and so the external specifications of SHRMAP will be described.

The implementation provided should, however, be general enough for most applications.

SHRMAP is first called from subroutine INITL with NC = -1. The value NC = -1 indicates that this is an initialization call to read data or perform any other task which may be necessary. No data values are expected to be returned with this call. SHRMAP is called from subroutine SHOREL with NC = 0 and NSHOR specified and a value for NP is returned. When NSHOR specifies one of the shorelines (and NC = 0), then the number of points defining that shoreline is returned in NP. This value is used as a limit on a loop index. When NSHOR is out of range, the value NP = 0 is returned which effects an exit from a loop over all of the disjoint shorelines.

The most frequent call to SHRMAP is from subroutines SHOREL/BNDREL and CENWID with NC = +1 and with NSHOR and NP specified. The call with this value of NC returns the X- and Y-coordinates of the point NP on shoreline NSHOR.

In the nominal version of SHRMAP provided, the maximum number of distinct shorelines is determined by the dimension of the arrays NSPTS and NSTRT. The maximum (total) number of shoreline points is determined by the dimension of the arrays XS and YS. The entries in the array NSPTS

indicate the number of points on each shoreline, while the entries in the array NSTRT provide base registers for indexing the arrays XS and YS to find the coordinates of the NP-th point.

4. INPUT DATA FOR PREPR2

In program PREPR2, all data is read in three of the first four program units entered. The data cards will now be described in the order required. A complete data set listing is shown in Table 1. Since the number of cards required will vary with the application, the description to follow will at times refer to groups of cards. We indicate the variables read, the FORTRAN format used, and the program unit in which the card or group of cards is read.

<u>Variables Read</u>	<u>Format</u>
1. TITL	(20A4)
This card read in program MAIN contains a title of up to 80 characters. The title is used to identify the printed and punched output and is reproduced as the first card of the output data set.	
2. NXREGN, NYREGN, IFPNCH, IFDLST, IFTRCE	(5I5)
This card is read in program MAIN. NXREGN specifies the number of columns of elements and NYREGN specifies the number of rows of elements. The next three entries are optional. If these entries are zero or blank, a data deck will be punched (IFPNCH), a listing of the data set will appear with the printed output (IFDLST) and a record of the intermediate results will be written or traced for diagnostic purposes (IFTRCE). If an integer other than zero is entered for these variables, the corresponding function will be inhibited.	
3. X(1), X(2), X(3), ... X(NXREGN+1)	(8F10.1)
These entries are the x-coordinates of the edges of the elements or the x-coordinates of the lines separating the internal elements. These are read in subroutine INITL.	

Variables Read	Format
4. Y(1), Y(2), ... Y(NYREGN+1)	(8F10.1)
These entries are the y-coordinates of the edges of the internal elements or of the lines separating those elements. These entries are read in subroutine INITL. The remaining entries define the shoreline and are read in subroutine SHRMAP called from subroutine INITL.	
5. NSHRMX	(I5)
This entry read in subroutine SHRMAP indicates the number of disjoint shorelines defining the water-land interface and specifies the number of repetitions of the next two data entries.	
6. NPTS	(I5)
This entry is the number of points defining an arc of a shoreline and the number of pairs of coordinates to be read.	
7. XS(1), YS(1), ... , XS(NPTS), YS(NPTS)	(2F10.0)
These are x- and y-coordinates defining the end points of the straight line segments which approximate the shoreline. A pair of coordinates appears on each card image.	

.

.

.

The data entries described in 6 and 7 are repeated NSHRMAX times.

For batch processing use where the data is entered from cards, the format used in data entry 7 results in a large number of cards. In such cases the format could easily be changed to (8F10.0) with four pairs of coordinates entered per card. In the application of PREPR2 at ORNL the data card images reside on disk storage where the single datum per card format is easier to edit and maintain.

Table 1. Input Data to PREPR2 for the Atlantic Generating Station Site (cont.)

20400.0	43400.0
21900.0	43600.0
22450.0	43600.0
23950.0	43600.0
25000.0	43600.0
25250.0	43600.0
25300.0	43600.0
25400.0	43600.0
25500.0	43600.0
25600.0	43600.0
25700.0	43600.0
25800.0	43600.0
25900.0	43600.0
26000.0	43600.0
26100.0	43600.0
26200.0	43600.0
26300.0	43600.0
26400.0	43600.0
26500.0	43600.0
26600.0	43600.0
26700.0	43600.0
26800.0	43600.0
26900.0	43600.0
27000.0	43600.0
27100.0	43600.0
27200.0	43600.0
27300.0	43600.0
27400.0	43600.0
27500.0	43600.0
27600.0	43600.0
27700.0	43600.0
27800.0	43600.0
27900.0	43600.0
28000.0	43600.0
28100.0	43600.0
28200.0	43600.0
28300.0	43600.0
28400.0	43600.0
28500.0	43600.0
28600.0	43600.0
28700.0	43600.0
28800.0	43600.0
28900.0	43600.0
29000.0	43600.0
29100.0	43600.0
29200.0	43600.0
29300.0	43600.0
29400.0	43600.0
29500.0	43600.0
29600.0	43600.0
29700.0	43600.0
29800.0	43600.0
29900.0	43600.0
30000.0	43600.0
30100.0	43600.0
30200.0	43600.0
30300.0	43600.0
30400.0	43600.0
30500.0	43600.0
30600.0	43600.0
30700.0	43600.0
30800.0	43600.0
30900.0	43600.0
31000.0	43600.0
31100.0	43600.0
31200.0	43600.0
31300.0	43600.0
31400.0	43600.0
31500.0	43600.0
31600.0	43600.0
31700.0	43600.0
31800.0	43600.0
31900.0	43600.0
32000.0	43600.0
32100.0	43600.0
32200.0	43600.0
32300.0	43600.0
32400.0	43600.0
32500.0	43600.0
32600.0	43600.0
32700.0	43600.0
32800.0	43600.0
32900.0	43600.0
33000.0	43600.0
33100.0	43600.0
33200.0	43600.0
33300.0	43600.0
33400.0	43600.0
33500.0	43600.0
33600.0	43600.0
33700.0	43600.0
33800.0	43600.0
33900.0	43600.0
34000.0	43600.0
34100.0	43600.0
34200.0	43600.0
34300.0	43600.0
34400.0	43600.0
34500.0	43600.0
34600.0	43600.0
34700.0	43600.0
34800.0	43600.0
34900.0	43600.0
35000.0	43600.0
35100.0	43600.0
35200.0	43600.0
35300.0	43600.0
35400.0	43600.0
35500.0	43600.0
35600.0	43600.0
35700.0	43600.0
35800.0	43600.0
35900.0	43600.0
36000.0	43600.0
36100.0	43600.0
36200.0	43600.0
36300.0	43600.0
36400.0	43600.0
36500.0	43600.0
36600.0	43600.0
36700.0	43600.0
36800.0	43600.0
36900.0	43600.0
37000.0	43600.0
37100.0	43600.0
37200.0	43600.0
37300.0	43600.0
37400.0	43600.0
37500.0	43600.0
37600.0	43600.0
37700.0	43600.0
37800.0	43600.0
37900.0	43600.0
38000.0	43600.0
38100.0	43600.0
38200.0	43600.0
38300.0	43600.0
38400.0	43600.0
38500.0	43600.0
38600.0	43600.0
38700.0	43600.0

5. RESULTS OF AN APPLICATION OF PREPR2: ATLANTIC GENERATING STATION

The two principal data components which define the discrete element structure are a description of a mesh defining a nominal element size and position and a digital map of the shoreline expressed in the coordinate system of the mesh. In the vicinity of the area to be simulated, this shoreline may consist of several disjoint areas each approximated by a connected set of line segments. Each set of segments is specified by the sequence of end points of the line segments taken such that the land lies to the right as that sequence is traversed in order.

The site and vicinity of the proposed Atlantic Generating Station (AGS), located 2.8 miles offshore opposite Little Egg Inlet, 11 miles northeast of Atlantic City, New Jersey, offers a fairly complex geometry and exercises many of the features of PREPR2. The element structure used has 900 elements arranged in a 30×30 grid, 65,000 feet on a side. The shore, which has been simplified, is represented by 210 line segments forming four disjoint shorelines, one of which is an island. A map of the shoreline drawn from this data is shown in Fig. 5. The same shoreline with the grid structure superimposed is shown in Fig. 6.

The input data for the PREPR2 code is shown in Table 1. The first line is the title, the next nine lines specify the grid, and the remainder define the shoreline.

A partial output listing produced by PREPR2 is reproduced in Appendix D with the options "TRACE" and "LPRINT" selected, and a complete listing on the microfiche at the end of this report. The first set of output which is always written is simply a repeat of the input data in a more compact format. This is followed by current information on the boundary

Fig. 5. Map of Shoreline Near Proposed Atlantic Generating Station.

Fig. 6. Map of Shoreline Near Proposed Atlantic Generating Station with Grid Structure Superimposed.

element being visited as each shoreline is traced in subroutines SHOREL and BNDREL. For each entry into an element, two or three types of information are normally printed: 1) terminal point if a line segment terminates in the element, 2) the side and point of exit, and 3) the cumulative area, centroids, and edge conditions at the event of exit. The sides of the element are numbered counterclockwise with the bottom side numbered 1. All warning and error messages are printed regardless of the trace option. Note for future reference the warning of disconnected water areas found in BNDREL.

MARKEX is called followed by a call to CENWID which completes each of the boundary elements found by SHOREL/BNDREL. The centroid widths found in CENWID are printed; then the indices of the small boundary elements deleted by REMSBE are printed. When a boundary element is deleted, the following elements are pushed up to fill the space left by its removal. This results in a renumbering of the boundary elements following the deleted one.

Finally, those elements which are boundary elements by virtue of lying around the open edge of the water area are identified and their properties printed by BNDFIN. At this point all of the necessary information to produce the output data set has been gathered. Before writing this data, two tables are printed.

The first table identifies each of the elements by type. An entry is printed at the appropriate place in the array of elements. An entry of -1 signifies that the element is of type external, an entry of 0 that the element is of type internal, and a positive entry that the element is a boundary element with the positive number identifying the (possibly renumbered) location in the array of boundary elements.

The second table lists each element according to type with a letter entry: external (E), internal (I), and boundary (B). Each boundary element (B) is surrounded by four entries which specify the edge conditions on that element. These conditions and entries are: adjacent to external element (E), adjacent to an internal or boundary element (I), open flow boundary conditions used (O), and specified flow boundary conditions to be used (S).

The final information printed is a listing of the data set prepared to be punched out on cards. This data may be used to partially reconstruct the original site geometry. Data set preparation is an averaging process where all of the details of a shoreline within an element are condensed into half a dozen numbers and even the approximate inversion of this process is not always possible. This is true when multiple shorelines enter an element and particularly when the shorelines produce disconnected water areas. In this latter case the description of such elements is not meaningful to FAROUT without further approximation and is indicative of a grid structure which is too coarse in the region of the element in question.

In Fig. 7 we present a reconstruction of the site geometry of AGS produced by the graphics program GRAFAR 4. This figure may be compared with Fig. 6. In Fig. 7 we see essential difficulties in the reconstruction of fifteen shaded elements involving reentrant shorelines and disconnected water areas. While most of these problems could be alleviated by suitable modifications to the subroutines SHOREL/BNDREL and REMSBE, the present version of PREPR2 requires that the data relating to these elements be corrected by hand if the present mesh is to be used. PREPR2 may still be used to advantage in this case as some 650 data cards are correct.

Fig. 7. Reconstruction of Shoreline Geometry by the Program GRAFAR
From a Data Set Produced by PREPR2.

REFERENCES

1. Arsev H. Eraslan, "A Transient, Two-Dimensional, Discrete-Element, Far-Field Model for Thermal Impact Analysis of Power Plant Discharges in Coastal and Offshore Regions. Part 1: General Description of the Mathematical Model and the Results of an Application," ORNL-4940 (1975).
2. J. M. Barton, "FAROUT: Input Instructions for the Computer Code for Thermal Analysis of Power Plant Effluents," (private communication).
3. Arsev H. Eraslan, et al., "FARTMP: A Tidal-Transient, Two-Dimensional Computer Code for Simulating Temperature Conditions in Coastal Regions, Lakes, and River Sections," (ORNL report in preparation).
4. B. J. Thomas, J. L. Bledsoe, and J. T. Holdeman, "GRAFAR: A Computer Code for Graphic Display of Input Data to the FAR2D/FAROUT Code," ORNL/CSD/TM-22, (1977).

APPENDIX A

The Computation of the First Moments of a Closed Curve with Particular Application to a Polygon

**The Computation of the First Moments of a Closed Curve
with Particular Application to a Polygon**

Consider the closed curve and the concave region enclosed by the curve shown below.

The area enclosed by the curve can obviously be calculated by

$$A = - \int_{x_2}^{x_1} y_2(x) dx - \int_{x_1}^{x_2} y_1(x) dx .$$

The first moment about the y-axis can be calculated from the integral

$$M_y = \int_{x_1}^{x_2} x(y_2(x) - y_1(x)) dx = - \int_{x_2}^{x_1} xy_2(x) dx - \int_{x_1}^{x_2} xy_1(x) dx .$$

In an analogous manner, the moment about the x-axis can be calculated from an integral of the form

$$M_x = \int_{y_1}^{y_2} y(x_2(y) - x_1(y)) dy = \int_{y_1}^{y_2} yx_2(y) dy + \int_{y_2}^{y_1} yx_1(y) dy .$$

For non-concave areas, the integrals and limits become more complicated. The integrals (particularly in the non-concave case) can be expressed more simply if the curve is expressed in parametric representation with parameter t :

$$x = x(t) , y = y(t) , a \leq t \leq b ,$$

where

$$x(a) = x(b) \text{ and } y(a) = y(b) .$$

Then

$$A = - \int_a^b y(t)x'(t)dt = \int_a^b x(t)y'(t)dt = - \oint y(t)x'(t)dt ,$$

where $x'(t) = d/dt x(t)$ and $y'(t) = d/dt y(t)$, and we assume that t increases as the curve is traversed in a counterclockwise sense. In a similar manner, the integrals for the moments become

$$M_x = \oint y(t)x(t)y'(t)dt$$

and

$$M_y = - \oint x(t)y(t)x'(t)dt .$$

The centroids of the enclosed area are given by

$$\bar{x} = M_x/A , \bar{y} = M_y/A .$$

Application to polygons.

Segments of the polygonal boundary curve may be parameterized as

$$\begin{aligned} x(t) &= x_1 + (x_2 - x_1)t \\ y(t) &= y_1 + (y_2 - y_1)t \end{aligned} \quad \begin{array}{l} t=1 \\ \nearrow \\ (x_2, y_2) \\ t=0 \swarrow \\ (x_1, y_1) \end{array}$$

The various integrals needed are

$$\begin{aligned} I_A &= - \int_0^1 y(t) \dot{x}(t) dt = - \int_0^1 [y_1 + (y_2 - y_1)t] (x_2 - x_1) dt \\ &= 1/2(y_1 + y_2)(x_1 - x_2) = 1/2(x_1 y_2 - x_2 y_1) + 1/2(x_1 y_1 - x_2 y_2) , \end{aligned}$$

$$\begin{aligned} I_x &= - \int_0^1 y(t) x(t) y'(t) dt \\ &= - \int_0^1 [y_1 + (y_2 - y_1)t] [x_1 + (x_2 - x_1)t] (y_2 - y_1) dt \\ &= 1/6(x_1 y_2 - x_2 y_1)(y_1 + y_2) + 1/3(x_2 y_2^2 - x_1 y_1^2) , \end{aligned}$$

and

$$\begin{aligned} I_y &= \int_0^1 y(t) x(t) \dot{x}(t) dt \\ &= 1/6(x_1 y_2 - x_2 y_1)(x_1 + x_2) - 1/3(y_2 x_1^2 - y_1 x_1^2) . \end{aligned}$$

When summing over the sides of a polygon, the homogeneous terms cancel and so are dropped in the sums to follow. Summing over the N sides of the polygon yields the results

$$\begin{aligned} A &= 1/2 \sum_{i=1}^N (x_i y_{i+1} - x_{i+1} y_i) , \\ M_x &= 1/6 \sum_{i=1}^N (x_i y_{i+1} - x_{i+1} y_i) (y_i + y_{i+1}) , \\ M_y &= 1/6 \sum_{i=1}^N (x_i y_{i+1} - x_{i+1} y_i) (x_i + x_{i+1}) , \end{aligned}$$

where the vertices are taken in counterclockwise order and where

$$x_{N+1} = x_1 \text{ and } y_{N+1} = y_1 .$$

For polygons these formulas have simple interpretations. To illustrate this, consider a triangle, the vertices of which are given by the vectors $\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_3$.

Note that the area of the triangle in the figure bounded by $\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_1 - \vec{r}_2$ is given by the z-component of the cross product,

$$A_{12} = 1/2[\vec{r}_1 \times \vec{r}_2]_z = 1/2(x_1 y_2 - y_1 x_2),$$

where the components of the vectors are given by

$$r_{ix} = x_i \text{ and } r_{iy} = y_i.$$

Then the area of the triangular region is given by

$$A = 1/2[\vec{r}_1 \times \vec{r}_2 + \vec{r}_2 \times \vec{r}_3 + \vec{r}_3 \times \vec{r}_1]_z = \sum_{i=1}^3 (x_i y_{i+1} - y_i x_{i+1}).$$

To illustrate the moment equation, note that the centroid of the triangle formed by vectors $\vec{r}_1$, $\vec{r}_2$, and $\vec{r}_1 - \vec{r}_2$ is given by

$$\vec{r}_c = 2/3 \cdot 1/2(\vec{r}_1 + \vec{r}_2),$$

and the moments are obtained by multiplying or weighting the components of this centroid by the area of the triangle,

$$\vec{M}_{12} = 1/3(\vec{r}_1 + \vec{r}_2) \cdot 1/2[\vec{r}_1 \times \vec{r}_2]_z .$$

Finally, the moments of the desired triangular region are given by the components of the moment vector $\vec{M}$,

$$\vec{M} = \sum_{i=1}^3 1/3(\vec{r}_i + \vec{r}_{i+1}) \cdot 1/2[\vec{r}_i \times \vec{r}_{i+1}]_z ,$$

where $\vec{r}_4 \equiv \vec{r}_1$. The formulas obtained previously result from expressing the vectors in terms of their components.

A practical problem arises in the direct application of these formulas when polygon is located some distance away from the origin of coordinates so that the lengths of the vectors involved are large compared with the dimensions of the polygon. The subtraction which is necessary in the computation of the components of the vector cross product results in a large decrease in significance. In this event the vectors should be referenced to a nearby coordinate system. The moments relative to the original coordinate system may be calculated by the parallel axis theorem while the area is of course invariant under such a transformation.

APPENDIX B
Program Listing of PREPR2

```

PROGRAM PREPARE STORAGE ALLOCATION
COMMON/PTITLE/PTITLE(20)
COMMON/PTPMP/LPUNCH,LPINT,TRACE,FILL(11)
COMMON/INOUT/IN,OUT,OUTOUT
INTEGER ULT,DATEUT
LOGICAL LPUNCH,LPINT,TRACE
DIMENSION APNCH(2),ADLST(2),ATRCE(2)
. . . . .

ARRAY S PROVIDES MOST OF THE STORAGE NEEDED FOR PROGRAM
NS IS DIMENSION OF ARRAY S

DIMENSION S(6100)
DATA NS/ 6100 /
. . . . . JTM . . . . .
DATA APNCH,ADLST,ATRCE/ 'P','NO F','D','NO D','T','N' //

INTEGER VARIABLE IN SPECIFIES THE LOGICAL INPUT UNIT.
INTEGER VARIABLE OUT SPECIFIES THE LOGICAL OUTPUT UNIT
USED FOR PRINTING, AND INTEGER VARIABLE OUTOUT SPECIFIES
THE LOGICAL UNIT USED FOR WRITING OR PUNCHING THE DATA
SET PRODUCED. LOGICAL VARIABLE TRACE=TRUE, ALLOWS AND
TRACE=FALSE, INHIBITS THE PRINTING OF DIAGNOSTIC INFO.

IN=05
OUT=06
OUTOUT=07
LPUNCH=.TRUE.
LPINT=.TRUE.
TRACE=.TRUE.
WRITE (OUT,1000)
1000 FORMAT(' F M E F A 2'ASK,'VERSION 7/76'/)
READ(IN,1010) TITLE
1010 FORMAT(2'A4)
READ(IN,1030) NXREGN,NYREGN,IFPACH,IFOLST,IFTRCE
1030 FORMAT(5I5)
IF(IFPACH.EQ.0) GO TO 10
IFPACH=1
LPUNCH=.FALSE.
10 IF(IFOLST.EQ.0) GO TO 20
IFOLST=1
LPINT=.FALSE.
20 IF(IFTRCE.EQ.0) GO TO 30
IFTRCE=1
TRACE=.FALSE.
30 WRITE (OUT,1015) APNCH(IFPACH+1),ADLST(IFOLST+1),ATRCE(IFTRCE+1)
1015 FORMAT(' OPTIONS: 'AX,A4,'PUNCH. 'A4,'ATA LIST. 'A4,'TRACE'/)
WRITE (OUT,1020) TITLE
1020 FORMAT(1X,20A4/)
WRITE (OUT,1040) NXREGN,NYREGN
1040 FORMAT(' NXREGN (NUMBER OF COLUMNS OF ELEMENTS) =',I3/
& ' NYREGN (NUMBER OF ROWS OF ELEMENTS) =',I3/)

ALGORITHMS FOR ASSIGNING LENGTHS TO ARRAYS
LELTY=NXREGN+NYREGN
LEND IS THE LENGTH ALLOCATED FOR BOUNDRY ELEMENT ARRAYS
LEND=3.5*(NXREGN+NYREGN)
NSEGN IS THE LENGTH ALLOCATED TO SEGMENT LIST FOR
BOUNDARY ELEMENTS
NSEGN=4*(NXREGN+NYREGN)
LSTACK IS LENGTH ALLOCATED FOR STACKS IN MARKER AND
BUFFERS IN PRINT
LSTACK=30*NP0(120,NXREGN+NYREGN+LELTY*5)

```

```

:
: GENERATE POINTERS FOR PARTITIONING STORAGE ARRAY S.
:
: ILTYP=1
: IX=ILTYP+LELTYF
: IY=IX+NYREGN+1
: ISIDE=IY+NYREGN+1
: IOSIDE=ISIDE+4*LBND
: IXC3=IOSIDE+4*LBND
: IJYC3=IXC3+LBND
: IBAR=IJYC3+LBND
: IBX3=IBAR+LBND
: IBY3=IBX3+LBND
: IXC3=IBY3+LBND
: IYCB=IXC3+LBND
: INREC=IYCB+LBND
: ILSG=INREC+LBND
: ISTACK=ILSG+4*NSEGN
: LAST=ISTACK+LSTACK-1
: IF (LAST.GT.NS) GO TO 100
: USE ALL OF REMAINING STORAGE FOR STACK
: LSTACK=MAX(LSTACK,((NS-ISTACK+1)/3)+3)
:
: CALL PREPH2 (NXREGN,NYREGN,LBND,NSEGN,LSTACK,S(ILTYP),
: S(IX),S(IY),S(ISIDE),S(IOSIDE),S(IXC3),S(IJYC3),
: S(IBAR),S(IBX3),S(IBY3),S(IXCB),S(IYCB),S(INREC),
: S(ILSG),S(ISTACK))
:
: NLEF=NS-LAST
: WRITE (OUT,1050) LAST,ALFT
: 1050 FORMAT (' ALLOCATCH = ',IS,' WORDS NECESSARY, ',
: 15,' WORDS EXCESS')
: STOP
:
: 100 WRITE (OUT,1060) NXREGN,NYREGN,LAST,LAST
: 1060 FORMAT ('// ' ## STORAGE ALLOCATION OVERFLOW ## /
: 15,' TO PREPARE A DATA SET FOR CASE NXREGN=',I3,' AND NYREGN='
: 15,' DIMENSION ARRAY S TO S(',IS,') AND SET NS=',IS)
: STOP
: EN)

```

```

C UNLOCATE FHEPZ INREGN, NYREGN, LEND, NSEGN, LSTK, IELTYP,
C X,Y, ISIDE, OSIDE, IXCE, JYCE, PAR, PAM, BYM, DXCE, DYCB, NBREC,
C LSEG, STAK)

C DIMENSION IELTYP(NXREGN, NYREGN), X(2), Y(2), ISIDE(4, LEND),
C OSIDE(4, LEND), IXCB(LEND), JYCB(LEND), PAM(LEND), PAM(LEND),
C BYM(LEND), JXCP(LEND), JYCP(LEND), NBRC(LEND),
C LSEG(4, NSEGN), STAK(LSTK)
COMMON/DTPREF/LPUNCH, LPRINT, THACE, LP, NJMO, NEXTNY, FIL(2)
COMMON/INOLT/IN, OUT, DATOUT
INTEGER OUT, DATOUT

NX=NXREGN+1
NY=NYREGN+1
NSTK=LSTK+1

CALL INITLIN(NXREGN, NYREGN, NXF, NYE, IELTYP, X, Y)

CALL SIGFIL(NXREGN, NYREGN, LEND, NSEGN, IELTYP, X, Y, ISIDE,
C OSIDE, IXCB, JYCB, PAR, EXP, BYM, NBREC, LSEG)

CALL MARKX(NXREGN, NYREGN, LEND, NSTK, IELTYP, ISIDE,
C IXCB, JYCB, STAK(1), STAK(NSTK+1), STAK(NSTK+1))

CALL CNMT(LEND, X, Y, ISIDE, IXCB, JYCB, PAM, EXP, BYM,
C NBRC, LSEG, JXCP, JYCB)

CALL REMSBE(NXREGN, NYREGN, LEND, IELTYP, X, Y, ISIDE, OSIDE,
C IXCB, JYCB, PAR, EXP, BYM, DXCE, DYCB, NBREC)

CALL INOTIN(NXREGN, NYREGN, LEND, IELTYP, X, Y, ISIDE, OSIDE,
C IXCB, JYCB, PAR, EXP, BYM, DXCE, DYCB)

CALL PRINTIN(NXREGN, NYREGN, LPAC, IELTYP, ISIDE, STAK(1),
C STAK(NSTK+1), STAK(NSTK+1))

IF(OUT.EQ.DATOUT) WRITE (OUT, 210)
210 FORMAT(11)

CALL DATDEX(NXREGN, NYREGN, LEND, IELTYP, X, Y, ISIDE, OSIDE,
C IXCB, JYCB, PAR, EXP, BYM, DXCE, DYCB)

RETURN
END

```

```

SUBROUTINE INITLINXREGN,NYREGN,NXE,NYF,IELTYP,X,Y)
C
C   DIMENSION IELTYP(NXREGN,NYREGN),X(NXF),Y(NYE)
COMMON/OTPHF/FILL(7),OXT,OYT,PAC(5)
COMMON/INCLT/IN,CLT,CATCUT
INTEGER OUT,OUTCUT
C
C   READ COORDINATES OF EDGES OF INTERNAL ELEMENTS
READ (IN,1010) X
READ (IN,1010) Y
1010 FORMAT(2F10.1)
C   NOW WRITE OUT THE COORDINATES OF MESH OF EDGES OF
C   INTERNAL ELEMENTS.
WRITE (OUT,1020)
1020 FORMAT(' NOMINAL ELEMENT EDGES',/,' N',6X,
6 'X-COORDINATES',5X,'Y-COORDINATES')
NI=MINO(NXF,NYE)
WRITE (OUT,1030) (1,X(I),Y(I),I=1,NI)
1030 FORMAT(1X,12,2F18.2)
IF(NXE.EQ.NYE) GO TO 20
NI=NI+1
IF(NYE.GT.NXE) GO TO 10
WRITE (OUT,1040) (1,X(I),I=NI,NXE)
1040 FORMAT(1X,12,F18.2)
GO TO 20
10 WRITE (OUT,1050) (1,Y(I),I=NI,NYE)
1050 FORMAT(1X,12,18X,F18.2)
C   INITIALIZE ALL ELEMENTS TO DEFAULT TO TYPE INTERNAL
20 DO 30 J=1,NYREGN
DO 30 I=1,NXREGN
30 IELTYP(I,J)=C
C   NOW CALL TO INITIALIZE THE SHORELINE SUBROUTINE
CALL SHRMAP(ASHCR,NF,XX,YY,-1)
C   COMPUTE SEGMENT RESOLUTION FOR BNDWEL
)XT=X(2)-X(1)
DO 40 I=2,NXE
40 OXT=AMINI(OXT,X(I)-X(I-1))
OXT=.02/OXT
OYT=Y(2)-Y(1)
DO 50 J=2,NYE
50 OYT=AMINI(OYT,Y(J)-Y(J-1))
OYT=.02/OYT
RETURN
END

```

```

SUBROUTINE SHCREL(NXREGN,NYREGN,LBND,NSGN,IE,TYP,X,Y,
& ISIDE,IXCB,JYCB,BAR,BXM,BYM,NBREC,LSEG)
...
SUBROUTINE TC TRACE SHCRELINE (CENT IFING BOUNDARY ELEMENTS
...
DIMENSION IELTYP(NXREGN,NYREGN),X(40),Y(40),ISIDE(40..BND),
& DSIDE(4,LBND),IXCB(LBND),JYCB(LBND),BAR(LBND),BXM(LBND),
& BYM(LBND),NBREC(LBND),LSEG(4,NSGN)
COMMON/DTPRFP/LPUNCH,LPRINT,TRACE,LP,NBND,NEXTAN,KIN,
& DXT,DYT,KOLT,XCUT,YCUT,IC,JC
COMMON/INOUT/IN,OUT,CATOUT
INTEGER OUT,CATOUT
LOGICAL LPUNCH,LPRINT,TRACE
DIMENSION IADD(4),JADD(4),ICC2(5),MX(5),MY(5)
DATA IADD/0.1,0.-1/,JADD/-1,0,1,0/,ICC2/3,4,1,2,3/
DATA MX/0.1,1,0,0/,MY/0,0,1,1,0/

LP=0
NBND=0

NSHOR=0
BEGIN LOOP OVER DISJOINT SHCRELINES
4 NSHOR=NSHOR+1
GET NUMBER OF POINTS NPTS ON SHCRELINE NSHOR
CALL SHMAP(NSHOR,NPTS,XB,YB)
IF(NPTS.EQ.0) GO TO 100
NP=1
6 IF(NP.GE.NPTS) GO TO 4
CALL SHMAP(NSHOR,NP,XP,YE,1)
XT=(X(1)-XB)*(X(NXREGN+1)-XB)
YT=(Y(1)-YB)*(Y(NYREGN+1)-YB)
IF(XT.GT.0.,.CR. YT.GT.0.) GO TO 22

FIRST POINT IS INSIDE ARRAY, FIND THE ELEMENT
IC=NXREGN+1
DO 8 I=1,NXREGN
IC=IC-1
8 IF(X(IC).LE.XB) GO TO 10
10 JC=NYREGN+1
DO 12 J=1,NYREGN
JC=JC-1
12 IF(Y(JC).LE.YB) GO TO 13
13 IF(TRACE) WRITE (OUT,1013) NSHOR,IC,JC,XB,YB
1010 FORMAT(/' SHOREL- SHCRELINE',12,' STARTS IN ELEMENT I=',
& 13,' J=',13,5X,'AT X,Y=',2F11.3/)
LOOP UNTIL A POINT OUTSIDE THE ELEMENT IS FOUND
14 IF(NP.GE.NPTS) GO TO 4
XA=XB
YA=YB
NP=NP+1
CALL SHMAP(NSHOR,NP,XP,YE,1)
IF(XB.LT.X(IC).CR. XB.GT.X(IC+1)) GO TO 16
IF(YB.LT.Y(JC).CR. YB.GT.Y(JC+1)) GO TO 16
GO TO 14

FOUND POINT OUTSIDE, FIND EXITING SIDE
16 DXA=XB-XA
DYA=YB-YA
DO 18 KK=1,4
XE=X(IC+MX(KK))
YE=Y(JC+MY(KK))
DXE=X(IC+MX(KK+1))-XE
DYE=Y(JC+MY(KK+1))-YE
CALL ICROSS(XA,DXA,YA,DYA,XE,DXE,YE,DYE,S,T)
IF(ABS(T-.5).GE..50001) GO TO 18
IF(ABS(S-.5).LE..50001) GO TO 20
18 CONTINUE
20 KOUT=KK
XOUT=XE+T*DXE
YOUT=YE+T*DYE
GO TO 6C
22 NPS=NP+1

```

```

C      DO 26 NN=APS,APTS
      GET NEXT POINT ON SHORELINE
      XA=X3
      YA=Y3
      NP=NN
      CALL SHRNAP(ASHCR,NP,XE,YE,1)
      DX=XB-XA
      DY=YB-YA
C      TEST FOR INTERSECTION OF SEGMENT WITH EACH EDGE OF MESH
      K=0
      SMIN=1.
      DO 24 KK=1,4
      XE=X(1+MX(KK)*NXREGN)
      YE=Y(1+MY(KK)*NYREGN)
      DXE=X(1+MX(KK+1)*NXREGN)-XE
      DYE=Y(1+MY(KK+1)*NYREGN)-YE
      CALL ICROSS(XA,DXA,YA,DYA,XE,DXE,YE,DYE,S,T)
      IF(ABS(T-.5).GE..50001) GC TO 24
      IF(ABS(S-.5).GE..50001) GC TO 24
      IF(S.GT.SMIN) GC TO 24
      SMIN=S
      TE=T
      K=KK
24  CONTINUE
      IF(K.GT.0) GC TO 30
26  CONTINUE
      WRITE (OUT,1011) ASHCR
1011 FORMAT(' SHCREL- SHCRELINE',I2,
        & ' NO FURTHER INTERSECTIONS WITH EDGE FOUND')
      GO TO 4
C
30  KOUT=ICC2(K)
      XE=X(1+MX(K)*NXREGN)
      XOUT=XE+TE*(X(1+MX(K+1)*NXREGN)-XE)
      YE=Y(1+MY(K)*NYREGN)
      YOUT=YE+TE*(Y(1+MY(K+1)*NYREGN)-YE)
      IF(TRACE) WRITE (CLT,1012) ASHCR,K,XOUT,YOUT
1012 FORMAT(' SHCREL- SHCFELINE',I2,' ENTERS SIDE',I2,
        & ' AT X,Y=',2F11,4/)
C      FIND INDICES OF THE ELEMENT WHICH SHORELINE ENTERS
      GO TO (40,60,40,60),K
40  JC=MY(K)*(NYREGN+1)
      IC=NXREGN+1
      DO 50 KK=1,NXREGN
      IC=IC-1
50  IF(X(IC).LT.XCLT) GC TO 60
      GO TO 50
60  IC=MX(K)*(NXREGN+1)
      JC=NYREGN+1
      DO 70 KK=2,NYREGN
      JC=JC-1
70  IF(Y(JC).LT.YCLT) GC TO 80
      JC=JC-1
C      END OF BLOCK TO FIND ENTRY POINT OF SHORELINE
C
C      SHORELINE SEGMENT ENTERS ELEMENT (IC,JC)
80  KIN=ICC2(KOUT)
      IC=IC+IADD(KCLT)
      JC=JC+JADD(KCLT)
C      CHECK IF SHORELINE LEAVES THE MESH
      IF(IC.LT.1 .OR. IC.GT.NXREGN) GC TO 6
      IF(JC.LT.1 .OR. JC.GT.NYREGN) GC TO 6
C
      CALL BNDREL(NXREGN,NYREGN,.BND,NSEGN,IELTYP,X,Y,ISIDE,
        & OSIDE,IXCB,JYCB,PAF,BXA,BYM,NBREC,LS EG,NSHOR,NPTS,NP,IFQ)
C
      IF(I=GO.EQ.0) GC TO 80
      GO TO 4
C
100 RETURN
      END

```



```

SUBROUTINE ENDREL(NXREGN,NYREGN,LBND,NSEGN,IELTYP,X,Y,ISIDE,
C DSIDE,IXCB,JYCB,EAF,BXN,BYN,NBREC,LSEG,NSHOR,NPTS,NP,IFQ)

```

```

C DIMENSION IELTYP(NXREGN,NYREGN),X(40),Y(40),ISIDE(4,-BND),
C DSIDE(4,LBND),IXCB(LBND),JYCB(LBND),EAF(LBND),BXN(LBND),
C BYN(LBND),NBREC(LBND),LSEG(4,NSEGN),RSIDE(2)
C COMMON/DTPREP/LPUNCH,LPRINT,TRACE,LP,NBND,NEXTNM,KIN,
C CXT,CYT,XOUT,XCLT,YCUT,IC,JC
C COMMON/INOUT/IN,CLT,OUT
C INTEGER OUT,DATELT
C LOGICAL LPUNCH,LPRINT,TRACE
C DIMENSION ICC(5),ICW(5),MX(5),MY(5),MS(5)
C EQUIVALENCE (RSIDE(1),OXR),(RSIDE(2),CYR)
C DATA ICC/2,3,4,1,2/, ICW/4,1,2,3,4/
C DATA MX/1,1,2,2/, MY/0,1,1,3/, MS/1,2,1,2,1/

```

```

IFQ=0
KFIRST=KIN
XL=X(IC)
XR=X(IC+1)
YL=Y(JC)
YR=Y(JC+1)
JX=XR-XL
JY=YR-YL
AR=AR+JX*JY
XIN=XOUT
YIN=YOUT
NB=XIN
NB=YIN
AKA=0
AKM=0
VAKM=0

```

```

C TEST FOR REENTRY OF EXISTING BOUNDARY ELEMENT
IF(IELTYP(IC,JC).EQ.0) GO TO 86
C INITIALIZATION FOR REENTRY OF EXISTING BOUNDARY ELEMENT
NB=IELTYP(IC,JC)
C FIND END OF ELEMENT SEGMENT LIST
NLP=NBREC(NB)

```

```

87 IF(LSEG(4,NLP).EQ.0) GO TO 84
NLP=LSEG(4,NLP)
GO TO 82

```

```

C NOW ADD NEW REFERENCE
84 LP=LP+1
IF(LP.GT.NSEGN) GO TO 220
LSEG(4,NLP)=LP
LSEG(1,NLP)=NSHOR
LSEG(2,NLP)=NP-1
LSEG(3,NLP)=NP-1
LSEG(4,NLP)=0
GO TO 90

```

```

C INITIALIZATION FOR NEW BOUNDARY ELEMENT
86 IF(NBND.GE.LBND) GO TO 230
NBND=NBND+1
NB=NBND
IELTYP(IC,JC)=NB
BAR(NB)=0
BXN(NB)=0
BYN(NB)=0
IXCB(NB)=IC
JYCB(NB)=JC

```

```

C ALL SURFACES INITIALLY MARKED EXTERNAL
GO 89 K=1,4
ISIDE(K,NB)=1
88 DSIDE(K,NB)=0
LP=LP+1
IF(LP.GT.NSEGN) GO TO 220
NBREC(NB)=LP
LSEG(1,NB)=NSHOR
LSEG(2,NB)=NP-1
LSEG(3,NB)=NP-1
LSEG(4,NB)=0

```

```

90 XA=XR
YA=YR
CALL SHMAP(NSHOR,NP,XB,YB,1)
92 IF(XL-XB.GT.DXT.(R-XR-XF.GT.DXT)) GO TO 95

```

```

      IF (YL-YB.GT.DYT .CR. YE-YU.GT.DYT) GO TO 95
      IF (NP.EQ.NPTS) GO TO 93
      = NO POINT POSSIBLY ON BOUNDARY. MOVE IT OFF
      IF (XB-XL.LT.DYT) XB=XE+SIGN(DYT,XE-XL)
      IF (XB-XB.LT.DYT) XB=XE-SIGN(DYT,XE-XB)
93  IF (XB.LE.XL .OR. XB.GE.XR) GO TO 95
      IF (NP.EQ.NPTS) GO TO 94
      IF (YB-YL.LT.DYT) YB=YE+SIGN(DYT,YE-YL)
      IF (YU-YB.LT.DYT) YB=YE-SIGN(DYT,YU-YB)
94  IF (YB.LE.YL .CR. YB.GE.YU) GC TO 95
      GO TO 105
95  DX=XB-XA
      DY=YB-YA
      K=KIN
C    CHECK EACH EDGE OF ELEMENT FOR INTERSECTION
      DO 100 KK=1,4
      K=ICM(K)
      XX=X(IC+MX(K))
      YY=Y(JC+MY(K))
      DX=X(IC+MX(K+1))-XX
      DY=Y(JC+MY(K+1))-YY
      CALL ICROSS(XA,DXA,YA,DYA,XX,DX,YY,DY,S,T)
      IF (S.LT..00001) GC TO 105
      IF (ABS(T-.5).GT..50001) GC TO 100
      IF (ABS(S-.50001).GT..5) GC TO 100
      IF (K.NE.KFIRST) GC TO 110
100  CONTINUE
      IF WE FALL THROUGH THIS LOOP WE ARE LOST
      GO TO 106
      = NO INTERSECTION. SEGMENT MUST TERMINATE INSIDE ELEMENT
      = FIND CONTRIBUTION TO AREA AND MOMENTS. THEN CONTINUE
      WITH NEXT SHORELINE SEGMENT
105  KFIRST=0
      IF (TRACE) WRITE (OUT,1013) NP,IC,JC,XE,YB
1013 FORMAT(' BNDREL- SEGMENT ENDING NP=',I3,' TERMINATES IN K=',
     & I3,' JC=',I3,' X,Y=',2F11.3)
      F=(XA-XL)*(YB-YL)-(XB-XL)*(YA-YL)
      AREA=AREA+F*.5
      XMOM=XMOM+F*(YA-YL+YB-YL)/6.
      YMOM=YMOM+F*(XA-XL+XB-XL)/6.
      NP=NP+1
      = LOOP BACK TO PICK UP NEXT SHORELINE SEGMENT IF POSSIBLE
      IF (NP.LE.NPTS) GC TO 90
      = CHECK AND FINISH ELEMENT IF CLOSED SHORELINE (I.E. AN
      = ISLAND OR LAKE) ELSE EXIT.
      CALL SHORAP(SHCR,1,XT,YT,1)
      IF (XT.EQ.XR .AND. YT.EQ.YE .AND. IFGO.EQ.0) GO TO 108
106  WRITE (OUT,1015) IC,JC
1015 FORMAT(' BNDREL- *** BOUNDARY ELEMENT I=',I3,' J=',
     & I3,' IN ERROR ***//,9X,' SHORELINE IS INCOMPLETE//')
      AREA=AREA+F
      XMOM=.5*(X(IC)+X(IC+1))*AREA
      YMOM=.5*(Y(JC)+Y(JC+1))*AREA
      IFGO=1
      GO TO 190
108  XA=XB
      XB=XT
      YA=YB
      YB=YT
      NP=1
      LSEG(3,LP)=LSEG(3,LP)+NPTS+1
      IFGO=1
      GO TO 92
C    SHORE SEGMENT EXITED ON SIDE K
110  XOUT=XA+S*DXA
      YOUT=YA+S*DYA
      XB=XOUT
      YB=YOUT
      KOUT=K
      F=(XA-XL)*(YB-YL)-(XB-XL)*(YA-YL)
      AREA=AREA+F*.5
      XMOM=XMOM+F*(YA-YL+YB-YL)/6.
      YMOM=YMOM+F*(XA-XL+XB-XL)/6.
      LSEG(3,LP)=NP-LSEG(3,LP)
      IF (K.NE.KIN) GC TO 150
      IF (DXA*(YIN-YOUT).GE.DYA*(XIN-XOUT)) GO TO 160

```

```

C      LOOP AROUND RECTANGLE TO ENTERING SIDE
150  XA=X3
    YA=Y3
    XB=X(1C+NX(K+1))
    YB=Y(1C+NY(K+1))
    F=(XA-XL)*(YB-YL)-(XB-XL)*(YA-YL)
    AREA=AREA+F*.5
    XMOM=XMOM+F*(YA-YL+YB-YL)/6.
    YMOM=YMOM+F*(XA-XL+XB-XL)/6.
    DSIDE(K,NB)=DSIDE(K,NB)+ABS((XB-XA)+(YB-YA))
    K=IC1(K)
    IF(K,NE,KLN) GC TO 150
C      HANDLE SIDE THROUGH WHICH SHORELINE ENTERED
160  XA=X3
    YA=Y3
    XB=XLN
    YB=YNL
    DSIDE(K,NB)=DSIDE(K,NB)+ABS((XB-XA)+(YB-YA))
    F=(XA-XL)*(YB-YL)-(XB-XL)*(YA-YL)
    AREA=AREA+F*.5
    XMOM=XMOM+F*(YA-YL+YB-YL)/6.
    YMOM=YMOM+F*(XA-XL+XB-XL)/6.
C      CORRECT FOR CRIGIN SHIFT
    XMOM=XMOM+YL*AREA
    YMOM=YMOM+XL*AREA
    IF(1/SEG(4,NBREC(NB)).EQ.0) GO TO 180
C      CORRECT FOR ENTRY OF ELEMENT
    XMOM=XMOM+RXN(NB)
    YMOM=YMOM+RYN(NB)
    AREA=AREA+BAR(NB)
    IF(AREA.GT.AREAF) GC TO 170
    ENTRY PRODUCED DISCONNECTED WATER AREAS
    WRITE (OUT,1018) IC,JC
1018 FORMAT(/' HNDREL- *** WARNING *** REENTRANT SHORE LINE'/
& 5X,'DISCONNECTED WATER AREAS IN BOUNDARY ELEMENT I,J=',2I4/)
    GO TO 180
C      MULTIPLE ENTRY OF ELEMENT MAY ACC MULTIPLES OF
C      RECTANGLE VALUES. REMOVE THESE MULTIPLES.
170  AREA=AREA-AREAF
    XMOM=XMOM-(Y(1C+1)+Y(1C))*.5*AREAF
    YMOM=YMOM-(X(1C+1)+X(1C))*.5*AREAF
    IC 175 KK=1.4
175  DSIDE(KK,NB)=APAXI(DSIDE(KK,NB)-RSIDE(NS(KK)),0.)
    IF(AREA.GT.AREAF) GC TO 170
180  DO 185 KK=1.4
    ISIDE(KK,NB)=-1
    IF(DSIDE(KK,NB).EQ.0.) ISIDE(KK,NB)=1
185  CONTINUE
190  BAR(NB)=AREA
    XM(NB)=XDCP
    YM(NB)=YMLN
    XCEN=RYM(NB)/AREA
    YCEN=BXN(NB)/AREA
    IF(.NOT. TRACE) GC TO 200
    WRITE (OUT,1014) NB,IC,JC,KCUT,XOUT,YOUT
1014 FORMAT(' BNDREL- EXIT ENTRY ELEM',I4,' I,J=',2I4,
& ' AT SIDE',I2,5X,' X,Y=',2F11,3)
    WRITE (OUT,1007) AREA,XCEN,YCEN
1007 FORMAT(9X,'AREA=',1F12,5,23X,'CENTROID X,Y=',2F11,3)
    WRITE (OUT,1016) (ISIDE(I,NB),I=1,4), (DSIDE(I,NB),I=1,4)
1016 FORMAT(9X,'ISIDE=',4I4,5X,'DSIDE=',4F11,3)
200  RETURN
220  WRITE (OUT,1020) LF
1020 FORMAT(/' BNDREL - SHORELINE POINTER LIST OVERFLOW. LP=',I4)
    STOP 1
230  WRITE (OUT,1030) NBND
1030 FORMAT(/' BNDREL - BOUNDARY LIST OVERFLOW. NBND=',I4,
& ' LARGER STORAGE ALLOCATION NEEDED')
    STOP 2
    END

```

```

SUBROUTINE MARKEX(NXREGN,NYREGN,LEND,NSTACK,IE,TYP,
& ISIDE,IXCB,JYCB,IS,JS,KS)

SUBROUTINE TO MARK EXTERNAL ELEMENTS
ALL ELEMENTS WITHIN A REGION ENCLOSED BY BOUNDARY
ELEMENTS ARE MARKED EXTERNAL.
NO INTERNAL ELEMENT MAY TOUCH AN EXTERNAL ELEMENT
EXCEPT AT A CORNER.
THE SEARCH AND MARK ALGORITHM IS IMPLEMENTED AS A
RECURSIVE PROCEDURE USING LOOPS AND A PUSHDOWN STACK.
REMARK: MISSING BOUNDARY ELEMENTS MAY CAUSE STACK OVERFLOW

COMMON/DTMPREP/LFUNCH,LFRENT,TRACE,LP,NBND,NEXTN,FL_(8)
COMMON/INOUT/IN,OUT,CATCUT
INTEGER OUT,DATECT
LOGICAL LPUNCH,LPRENT,TRACE
DIMENSION IXCB(LBND),JYCB(LBND),ISIDE(4,LBND),
1 IE(TYP(NXREGN,NYREGN),IS(NSTACK),JS(NSTACK),KS(NSTACK))
DIMENSION IADD(4),JADD(4),ICCI(5),ICW(5)
DATA IADD/0,1,0,-1/,JADD/-1,0,1,0/,ICCI/2,3,3,1,2/,
& ICW/4,1,2,3,4/
WRITE (OUT,1000)
1000 FORMAT(' MARKEX-')
C BEGIN LOOP ON BOUNDARY ELEMENTS
NEXTN=1
DO 90 NB=1,NBND
IF OFLO=0
II=IXCB(NB)
JJ=JYCB(NB)
C LOOP OVER SIDES LOOKING FOR ADJACENT EXTERNAL ELEMENT
DO 80 KK=1,4
IF (ISIDE(KK,NB).NE.1) GO TO 90
I=II+IADD(KK)
J=JJ+JADD(KK)
K=KK
NS=0
C BEGIN RECURSIVE PROCEDURE SECTION TO MARK EXTERNAL ELMTS
C ARE COMPLETED INDICES VALID ?
40 IF (I.LT.1 .OR. I.GT.NXREGN) GO TO 70
IF (J.LT.1 .OR. J.GT.NYREGN) GO TO 70
C IS ELEMENT MARKED BOUNDARY OR EXTERNAL ?
IF (IE(TYP(I,J),NE,0) GO TO 70
NO, SC MARK IT EXTERNAL
IE(TYP(I,J))=-1
NEXTN=NEXTN+1
C NOW CHECK ADJACENT ELEMENTS
K=ICW(K)
C IS STACK FULL ?
IF (NS.LE.NSTACK-2) GO TO 50
C A STACK OVERFLOW IS NOT USUALLY IN ITSELF A FATAL ERROR
C BUT OFTEN INDICATES AN ERROR OR INEFFICIENT DESIGN
IF (OFLO.GT.0) GO TO 60
WRITE (OUT,1010) I,J,II,JJ
1010 FORMAT(' MARKEX- STACK OVERFLOW AT ELEMENT I=',I3,
& ' J=',I3,' FROM FCCT I=',I3,' J=',I3)
IF OFLO=1
GO TO 60
C ADD ELEMENT REFERENCES TO STACK
50 NS=NS+1
IS(NS)=I+IADD(K)
JS(NS)=J+JADD(K)
KS(NS)=K
K=ICCI(K)

```

```

      NS=NS+1
      I(S(NS))=I+IADD(K)
      J(S(NS))=J+JADD(K)
      K(S(NS))=K
60    K=ICCI(K)
      I=I+IADD(K)
      J=J+JADD(K)
      GO TO 40
      NOB POP NEXT ELEMENT REFERENCE OFF OF STACK
      EXIT IF STACK IS EMPTY
70    IF (NS.EQ.0) GO TO 40
      I=I(S(NS))
      J=J(S(NS))
      K=K(S(NS))
      NS=NS-1
      GO TO 40
      END OF RECURSIVE PROCEDURE SECTION
80    CONTINUE
90    CONTINUE
      RETURN
      END

```

```

SUBROUTINE CENND(LHND,X,Y,ISIDE,IXCP,JYCE,BAR,BAM,PYM,
& NPREC,LSEG,DXCB,DYCB)
C
C SUBROUTINE TO CALCULATE THE FITTED WIDTH THROUGH THE
C CENTROID OF BOUNDARY ELEMENTS
C
COMMON/OTPHED/LPUNCH,LFRINT,TRALE,LP,NBND,NEXTNN,FIL(8)
COMMON/INOLT/IN,CLT,CATCUT
INTEGER IULT,DATELT
LOGICAL LPUNCH,LFRINT,TRACE
DIMENSION X(2),Y(2),ISIDE(4,LPND),IXCB(LPND),JYCB(LPND),
& BAR(LPND),BAM(LPND),BYCE(LPND),NPREC(LPND),LSEG(4,2),
& DXCB(LPND),DYCB(LPND)
C
IF(TRACE) WRITE(CUT,1000)
1000 FORMAT(/' CFABIC-')
NSHORL=-1
DO 10 I=1,NBND
IC=IXCB(NB)
JC=JYCB(NB)
XC=N*HYMN(NB)/BAR(NB)
YC=N*BXM(NB)/BAR(NB)
XC=X(IC)
YC=Y(JC)
DXC=X(IC+1)-XC
DYC=Y(JC+1)-YC
DXCB(NB)=0.
DYCB(NB)=0.
LP=NBREC(NB)
10 NSHORL=LSEG(1,LP)
IF(NSHORL.EQ.NSHORL) GO TO 20
CALL SHMAP(ASHCR,NPTS,XB,YB,0)
NSHORL=NSHORL
20 NP=LSEG(2,LP)
NMP=LSEG(3,LP)
SHMAP RETURNS THE COORDINATES OF POINT NP ON
SHINE LINE ASHCR
CALL SHMAP(ASHCR,NP,XB,YB,1)
DO 90 N=1,NMP
XA=XB
YA=YB
NP=NP+1
IF(NP.GT.NPTS) IF=2
CALL SHMAP(ASHCR,NP,XB,YB,1)
DXA=XB-XA
DYA=YB-YA
C
C FITTED X-BIDTH THROUGH THE CENTROID
50 CALL ICROSS(XA,DXA,YA,DYA,XC,DXC,YCEN,0.5,S,T)
IF(ABS(T-.5).GT..5) GO TO 70
IF(ABS(S-.5).GT..50001) GO TO 70
XCRS=X+T*DXC
CROSS PRODUCT SELECTS APPROPRIATE LENGTH
IF(-DYA*DXC.GT.0) GO TO 60
DXCB(NB)=(XCRS-XC)+DXCB(NB)
GO TO 70
60 DXCB(NB)=(X(IC+1)-XCRS)+DXCB(NB)
C
C FITTED Y-BIDTH THROUGH THE CENTROID
70 CALL ICROSS(XA,DXA,YA,DYA,XCEN,0.5,YC,DYCB,S,T)
IF(ABS(T-.5).GT..5) GO TO 90
IF(ABS(S-.5).GT..50001) GO TO 90
YCRS=Y+T*DYC
CROSS PRODUCT SELECTS APPROPRIATE LENGTH
IF(DXA*DYC.GT.0) GO TO 80
DYCB(NB)=(YCRS-YC)+DYCB(NB)
80
90

```

```

      GO TO 90
      NO DYCB(NB)=(Y(JC+1)-YCS)+DYCB(NB)
      SC CONTINUE
100 LP=LSEG(4,LP)
      IF(LP.GT.0) GC TC 10
      REMOVE MULTIPLES OF RECTANGLE WIDTH WHICH MAY HAVE
      ACCUMULATED DUE TO SHORELINE RE-CROSSING CENTROID
110 IF(DYCB(NB).LE.DXC) GC TO 115
      DXC(NB)=DXC(NB)-DXC
      GO TO 110
115 IF(DYCB(NB).LE.0.) DXC(NB)=DXC
      REMOVE MULTIPLES OF RECTANGLE HEIGHT
120 IF(DYCB(NB).LE.DYC) GC TC 125
      DYCB(NB)=DYCB(NB)-DYC
      GO TO 120
125 IF(DYCB(NB).LE.0.) DYCB(NB)=DYC
      IF(TRACE) WRITE (OUT,1001) A6.1C,JC,DXC(NB),DYCB(NB)
1001 FORMAT(9X,'M='',I3,'',I3='',I3,'')
      L * CENTROID WIDTHS X,Y='',2F11.3)
130 CONTINUE
      RETURN
      END

```

```

SUBROUTINE REPSBE (XREGA, YREGA, LEND, IELTYP, X, Y, ISIDE,
C DSIDE, IXCB, JYCB, PAR, PER, BYR, ORCP, JYCB, NDBEC)

SUBROUTINE TO REMOVE SMALL BOUNDARY ELEMENTS FROM LIST
OF BOUNDARY ELEMENTS AFTER CONDITION ON THE INTEGRATION

DIMENSION IELTYP(XREGA, YREGA), X(2), Y(2), ISIDE(4, LEND),
C DSIDE(4, LEND), IXCB(LEND), JYCB(LEND), PAR(LEND), BYR(LEND),
C BYR(LEND), ORCP(LEND), JYCB(LEND), NDBEC(LEND)
DIMENSION ICCI(5), IADC(4), JADD(4)
COMMON/DTPREP/LPLACH, LPPINT, TRACE, LF, MND, NEXTN, FILL(8)
COMMON/INOLT/ IN, CUT, CATCUT
INTEGER ULT, OATCUT
LOGICAL LPLACH, LPPINT, TRACE
DATA IADC/0, 1, 0, -1/, JADD/-1, 0, 1, 0/, ICCI/2, 3, 4, 1, 2/
DATA FRAC/ .50/

      F=ND AREA OF SMALLEST INTERNAL ELEMENT
      IF (MND.LE.2) RETURN
      ARMIN=(X(2)-X(1))*(Y(2)-Y(1))*10000.
      ARMIN=ARMIN
      DO 20 JC=1, NDBEC
      JYR=Y(JC+1)-Y(JC)
      JC 10 IC=1, XREGA
      ARTRY=YR*(X(IC+1)-X(IC))
      IF (ARTRY.GE.ARMIN) GC TC 10
      IF (IELTYP(IC, JC).NE.0) GC TC 10
      ARMIN=ARTRY
10 CONTINUE
20 CONTINUE
      IF (ARMIN.EQ.ARMIN) RETURN

      EXAMINE EACH BOUNDARY ELEMENT AND DELETE THOSE BOUNDARY
      ELEMENTS WHICH HAVE AREA LESS THAN FRAC TIMES THE AREA OF
      THE SMALLEST INTERNAL ELEMENT. SUGGESTED VALUES FOR FRAC ARE
      1.0 = FRAC = .25

      ARTEST=FRAC*ARMIN
25 NUMCHG=0
      ND=1
30 IF (ND.GT.MND) GC TC 80
      EXAMINE SIDES OF EACH BOUNDARY ELEMENT COUNTING EXTERNAL
      EDGES (KE) AND FIND SIDE (KL) WITH LONGEST NETTED LENGTH
      KE=0
      KL=0
      BIGSID=0
      DO 35 K=1, 4
      IF (DSIDE(K, ND).NE.1) GC TC 33
      KE=KE+1
      GO TO 35
33 IF (DSIDE(K, ND).LT.BIGSID) GC TO 35
      BIGSID=DSIDE(K, ND)
      KL=K
35 CONTINUE
      TEST FOR SMALL ELEMENT
      IF (BAR(ND).LT.ARTEST .AND. KE.GT.0) GC TO 40
      TEST FOR ISOLATED BOUNDARY ELEMENT
      IF (KE.EQ.3) GC TC 40
      ND=ND+1
      GO TO 30
      FOUND SMALL ELEMENT, PREPARE TO DELETE IT
40 NUMCHG=NUMCHG+1
      IR=IXCB(ND)
      JR=JYCB(ND)
      MARK IT EXTERNAL
      IELTYP(IR, JR)=-1
      NEXTN=NEXTN+1
      WRITE (OUT, 1000) IR, JR
1000 FORMAT(' REPSBE- REMOVED SMALL BOUNDARY ELEMENT IC=',
C I4, ' JC=', I4)

```



```

C      CORRECT REFERENCES TO SURROUNDING ELEMENTS
DO 50 K=1,4
  IF (ISIDE(K,NB).EQ.1) GO TO 50
  IC=IR+IA*DO(K)
  IF (IC.LT.1 .OR. IC.GT.N*REGN) GO TO 50
  JC=JR+JA*DO(K)
  IF (JC.LT.1 .OR. JC.GT.N*REGN) GO TO 50
  NC=IELTYP(IC,JC)
  IF (NC.NE.0) GO TO 44
C      NUMBER INTERNAL ELEMENT BECOMES NEW BOUNDARY ELEMENT
  NBND=NBND+1
  NC=NBND
  IELTYP(IC,JC)=NC
  DO 44 KK=1,4
44  ISIDE(KK,NC)=-1
      XN=X(IC+1)-X(IC)
      YN=Y(JC+1)-Y(JC)
      AREA=0.5*DN
      XAX(NC)=AREA
      XCN(NC)=(Y(JC)+DN*0.5)*AREA
      YCN(NC)=(X(IC)+CN*0.5)*AREA
      IXCB(NC)=IC
      JYCB(NC)=JC
      OXCB(NC)=OXN
      OYCB(NC)=OYN
      DSIDE(1,NC)=OXN
      DSIDE(2,NC)=OYN
      DSIDE(3,NC)=OXN
      DSIDE(4,NC)=OYN
      IF (K.NE.KL) GO TO 45
      XCEN=0.5*(XCN(NC)+BAR(NC))
      YCEN=0.5*(YCN(NC)+BAR(NC))
      XBAR(NC)=0.5*(XCN(NC)+OXN(NC))
      YBAR(NC)=0.5*(YCN(NC)+OYN(NC))
      OXCB(NC)=OXCB(NC)+2.*ABS(YCN(NC)/BAR(NC)-XCEN)
      OYCB(NC)=OYCB(NC)+2.*ABS(XCN(NC)/BAR(NC)-YCEN)
      BAR(NB)=0.
      XBAR(NB)=0.
      YBAR(NB)=0.
C
44  KC=IC+1(K+1)
      ISIDE(KC,NC)=1
      DSIDE(KC,NC)=0.
      ISIDE(K,NB)=0
      DSIDE(K,NB)=0.
      KE=KE+1
50  CONTINUE
C      NOW DELETE REFERENCE AND COMPACT BOUNDARY
      ELEMENT LIST. WE WILL NOT ALTER THE ARRAY LSEG.
  NBND=NBND-1
  IF (NBND.NE.0) GO TO 60
  DO 70 N=NB,PRND
    NBREC(N)=NBREC(N+1)
    IXCB(N)=IXCB(N+1)
    JYCB(N)=JYCB(N+1)
    XBAR(N)=XBAR(N+1)
    YBAR(N)=YBAR(N+1)
    OXCB(N)=OXCB(N+1)
    OYCB(N)=OYCB(N+1)
  DO 60 K=1,4
    DSIDE(K,N)=DSIDE(K,N+1)
60  ISIDE(K,N)=ISIDE(K,N+1)
70  IELTYP(IXCB(N),JYCB(N))=N
  GO TO 30
80  IF (NUNCHG.NE.0) GO TO 25
  RETURN
END

```

```

SUBROUTINE BNOFIN(AXREGA,NYREGA,LBND,IELTYP,X,Y,ISIDE,DSIDE,
  6 IXC,JYC,BAR,BAP,BVP,DXC,DYCB)

  SUBROUTINE TO IDENTIFY AND MARK ADDITIONAL BOUNDARY
  ELEMENTS AROUND EDGE OF MESH
  PRINT TO ENTRY INTO BNOFIN, SUBROUTINES SHOWED AND
  MARKS HAVE BEEN CALLED SO ANY REMAINING ELEMENTS AROUND
  THE EDGE OF THE AREA MUST BE BOUNDARY ELEMENTS.

  DIMENSION IELTYP(AXREGA,NYREGA), ISIDE(4,LBND), DSIDE(4,LBND),
  6 IXC(LBND), JYC(LBND), RSIDE(2), KS(4), XLBND(1), YLBND(1),
  6 BXB(LBND), BYB(LBND), CXC(LBND), CYC(LBND)
  COMMON/DTPREP/LFLACH,LPRINT,TRACE,LP,NBND,NEXTN,FIL(8)
  COMMON/INOLT/IA,CLT,CATOUT
  INTEGER OLT,OLTCUT
  LOGICAL LFLACH,LPRINT,TRACE
  DATA KS/1,2,1,2/, MARKEDG/ 2/
  MARKEDG=0 FOR SPECIFIED FLOW, MARKEDG=2 FOR OPEN FLOW

  IF(TRACE) WRITE (CLT,1000)
1000 FORMAT(/
  NB1=NBND+1

  IDENTIFY UNMARKED BOUNDARY ELEMENTS ALONG UPPER AND
  LOWER BORDERS OF MESH
  OYI=Y(2)-Y(1)
  OYN=Y(NYREGA+1)-Y(NYREGA)
  DO 40 IC=1,AXREGA
    RSIDE(1)=X(IC+1)-X(IC)
    CHECK ELEMENT ON LOWER BORDER IN COLUMN IC
    LTST=IELTYP(IC,1)
    IF(LTST.LT.0) GO TO 20
    CHECK IF EXISTING BOUNDARY ELEMENT NEEDS TO BE MARKED OPEN
    IF(LTST.EQ.0) GO TO 10
    IF(ISIDE(1,LTST).EQ.-1) ISIDE(1,LTST)=MARKEDG
    IF(IC.NE.1) GO TO 5
    IF(ISIDE(4,LTST).EQ.-1) ISIDE(4,LTST)=MARKEDG
    GO TO 20
  5 IF(IC.NE.AXREGA) GO TO 20
    IF(ISIDE(2,LTST).EQ.-1) ISIDE(2,LTST)=MARKEDG
    GO TO 20
  10 MARK BOUNDARY ELEMENTS ALONG THE LOWER BORDER
  IF(NBND.GE.LBND) GO TO 95
  NBND=NBND+1
  IELTYP(IC,1)=NBND
  IXC(NBND)=IC
  JYC(NBND)=1
  DXC(NBND)=RSIDE(1)
  DYCB(NBND)=OYI
  RSIDE(2)=OYI
  DO 15 K=1,4
    ISIDE(K,NBND)=-1
  15 DSIDE(K,NBND)=RSIDE(KS(K))
    MARK OPEN SIDE
    ISIDE(1,NBND)=MARKEDG
    CORNER ELEMENTS HAVE EXTRA OPEN SIDE
    IF(IC.EQ.1) ISIDE(4,NBND)=MARKEDG
    IF(IC.EQ.AXREGA) ISIDE(2,NBND)=MARKEDG
    CHECK ELEMENT ON UPPER BORDER IN COLUMN IC
  20 LTST=IELTYP(IC,NYREGA)
    IF(LTST.LT.0) GO TO 40
    IF(LTST.EQ.0) GO TO 30
    CHECK IF EXISTING BOUNDARY ELEMENT NEEDS TO BE MARKED OPEN
    IF(ISIDE(3,LTST).EQ.-1) ISIDE(3,LTST)=MARKEDG
    IF(IC.NE.1) GO TO 25
    IF(ISIDE(4,LTST).EQ.-1) ISIDE(4,LTST)=MARKEDG
    GO TO 40
  25 IF(IC.NE.AXREGA) GO TO 40
    IF(ISIDE(2,LTST).EQ.-1) ISIDE(2,LTST)=MARKEDG
    GO TO 40
  30 MARK BOUNDARY ELEMENTS ALONG UPPER BORDER
  IF(NBND.GE.LBND) GO TO 95
  NBND=NBND+1
  IELTYP(IC,NYREGA)=NBND
  IXC(NBND)=IC
  JYC(NBND)=NYREGA
  DXC(NBND)=RSIDE(1)
  DYCB(NBND)=OYN
  RSIDE(2)=OYN
  DO 35 K=1,4

```

```

      ISIDE(K,NBND)=-1
35  JSIDE(K,NBND)=RSIDE(KS(K))
      MARK OPEN SIDE
      ISIDE(3,NBND)=MARKEDG
      CONNED ELEMENTS HAVE EXTRA OPEN SIDE
      IF(IC.EQ.1) ISIDE(4,NBND)=MARKEDG
      IF(IC.EQ.NXREGN) ISIDE(12,NBND)=MARKEDG
40  CONTINUE

      IDENTIFY UNMARKED BOUNDARY ELEMENTS ALONG THE LEFT
      AND RIGHT BORDERS OF THE MESH
      X=X(I2)-X(I1)
      Y=Y(NXREGN+1)-Y(NXREGN)
      NY=NXREGN-1
      DO 40 JC=2,NY
      JSIDE(2)=Y(JC+1)-Y(JC)
      CHECK ELEMENT ON LEFT BORDER IN ROW JC
      LTST=LTYP(1,JC)
      IF(LTST.LT.0) GO TO 60
      IF(LTST.EQ.0) GO TO 50
      IF(ISIDE(4,LTST).EQ.-1) ISIDE(4,LTST)=MARKEDG
      GO TO 60
      MARK BOUNDARY ELEMENTS ALONG THE LEFT BORDER
45  IF(NBND.EQ.LBND) GO TO 95
      NBND=NBND+1
      IF(LTYP(1,JC)=NBND)
      IX(NBND)=1
      JY(NBND)=JC
      JSIDE(1)=0X
      IX(NBND)=0X
      JY(NBND)=JSIDE(2)
      DO 50 K=1,4
      ISIDE(K,NBND)=-1
50  JSIDE(K,NBND)=RSIDE(KS(K))
      ISIDE(4,NBND)=MARKEDG
      CHECK ELEMENT ON RIGHT BORDER IN ROW JC
60  LTST=LTYP(NXREGN,JC)
      IF(LTST.LT.0) GO TO 80
      IF(LTST.EQ.0) GO TO 65
      IF(ISIDE(12,LTST).EQ.-1) ISIDE(12,LTST)=MARKEDG
      GO TO 40
      MARK BOUNDARY ELEMENTS ALONG THE RIGHT BORDER
65  IF(NBND.EQ.RBND) GO TO 95
      NBND=NBND+1
      IF(LTYP(NXREGN,JC)=NBND)
      IX(NBND)=NXREGN
      JY(NBND)=JC
      JSIDE(1)=0X
      IX(NBND)=0X
      JY(NBND)=RSIDE(2)
      DO 70 K=1,4
      ISIDE(K,NBND)=-1
70  JSIDE(K,NBND)=RSIDE(KS(K))
      ISIDE(12,NBND)=MARKEDG
80  CONTINUE
      IF(NBND.GT.NBND) RETURN
      ENTER AREA AND MOMENTS OF BORDER BOUNDARY ELEMENTS
      DO 90 N=NB1,NBND
      IC=IX(N)
      JC=JY(N)
      AREA=(X(IC+1)-X(IC))*Y(JC+1)-Y(JC)
      BAY(N)=AREA
      BYM(N)=(Y(JC+1)+Y(JC))*0.5*AREA
      BYX(N)=(X(IC+1)+X(IC))*0.5*AREA
      IF(.NOT.TRACE) GO TO 90
      XCEN=BYM(N)/AREA
      YCEN=BYX(N)/AREA
      WRITE (OUT,1010) N,IC,JC,AREA,XCEN,YCEN
1010 FORMAT(' BNDFA- NB=',I3,' I=',I2,' J=',I2,4X,
        & 'AREA=',1P12.5,' CENTRGDS X,Y=',0P2F11.3)
      WRITE (OUT,1020) (ISIDE(K,N),K=1,4),(JSIDE(K,N),K=1,4)
1020 FORMAT(4X,'ISIDE=',4I4,4X,'JSIDE=',4F11.3)
      GO TO 90
      RETURN
95  WRITE (OUT,1030)
1030 FORMAT(' BNDFA - BOUNDARY ELEMENT ARRAY OVERFLOW /
        & ' LARGER BOUNDARY STORAGE ALLOCATION NEEDED')
      RETURN
      END

```

```

SUBROUTINE PRINT(NXREGN,NYREGN,LEND,IELTYP,ISIDE,
& ROW1,ROW2,ROW3)
C
C SUBROUTINE TO DISPLAY ELEMENT TYPES AND EDGE CONDITIONS
C ON THE LINE PRINTER
C
C DIMENSION IELTYP(NXREGN,NYREGN),ISIDE(4,LEND),
& ELTYPE(3),SIDE(4),FCB1(128),ROW2(128),ROW3(128)
COMMON/PTI TLE/TITLE(20)
COMMON/INOUT/IN,OUT,OUT
INTEGER OUT,OUTCLT

DATA BLANK/1H /, ELTYPE/1HE,1HI,1HB/, SIDE/1P,1HS,1NE,1NW/

IMAX=MINO(32,NXREGN)
IMAX4=4*IMAX

11=1
12=IMAX
5 WRITE(OUT,1000) TITLE
1000 FORMAT(141,20A4/)
WRITE(OUT,1010)
1010 FORMAT(' IDENTIFICATION OF ELEMENTS BY THEIR INTERNAL INDICES'/)
JR=NYREGN
DO 10 JC=1,NYREGN
WRITE(OUT,1015) JR,(IELTYP(I,JR),I=11,12)
1015 FORMAT(1X,12,1H-,3214)
10 JC=JR-1
WRITE(OUT,1020) (I,I=11,12)
1020 FORMAT(7X,3214)
IF(12.EQ.0,NXREGN) GC TO 15
GO BACK AND PRINT RIGHT PORTION IF NXREGN.GT. 32
12=NXREGN
11=12-31
GO TO 5

15 11=1
12=IMAX
20 WRITE(OUT,1000) TITLE
WRITE(OUT,1025)
1025 FORMAT(' IDENTIFICATION OF ELEMENTS BY TYPE AND CONNECTIONS'/)
JR=NYREGN
DO 40 JC=1,NYREGN
C INITIALIZE THREE ROW BUFFERS TO BLANKS
DO 25 I=1,IMAX4
ROW1(I)=BLANK
ROW2(I)=BLANK
ROW3(I)=BLANK
25 ROW3(I)=BLANK
IR=3
DO 30 I=11,12
NB=IELTYP(I,JR)
ICHAR=MINO(1,NB)+2
ROW2(IR)=ELTYPE(ICHAR)
IF(NB.LE.0) GC TO 30
ROW1(IR)=SIDE(1SIDE(3,NB)+2)
ROW3(IR)=SIDE(1SIDE(1,NB)+2)
ROW2(IR-1)=SIDE(1SIDE(4,NB)+2)
ROW2(IR+1)=SIDE(1SIDE(2,NB)+2)
30 IR=IR+4
WRITE(OUT,1030) (FCB1(I),I=1,IMAX4)
1030 FORMAT(74X,128A1)
WRITE(OUT,1040) JR,(FCB2(I),I=1,IMAX4)
1040 FORMAT(1X,12,1X,128A1)
WRITE(OUT,1050) (FCB3(I),I=1,IMAX4)
1050 FORMAT(4X,128A1)
40 JR=JR-1
WRITE(OUT,1020) (I,I=11,12)
IF(12.EQ.0,NXREGN) GC TO 5
12=NXREGN
11=12-31
GO TO 20
50 RETURN
END

```



```

DO 70 J=1,NYREGN
DO 60 I=1,NXREGN
IF (IELTYP(I,J).LE.0) GO TO 60
NB=IELTYP(I,J)
NBN=NBN+1
AREA=BAR(NB)
XCN=BYN(NB)/AREA
YCN=BYX(NB)/AREA
NEW DATA FORMAT 5/76
WRITE (XCBN,XCBND,JCBN,DXBJPH,CXBJPH,
      JYCBND,YCBND,DYCBND,DYBJPH,
      IF(LPUNC1) WRITE(CATCLT,1050) IXCE(NB),XCEN,DXCB(NB),
      & DSIDE(1,NB),DSIDE(3,NB), JYCB(NB),YCBN,DYCB(NB),
      & DSIDE(4,NB),DSIDE(2,NB)
1050 FORMAT(8X,I3,4X,4(2X,E13.6)/8X,I3,4X,4(2X,E13.6))
      IF(LPINT1) WRITE(CUT,1055) IXCB(NB),XCEN,DXCB(NB),
      & DSIDE(1,NB),DSIDE(3,NB), JYCB(NB),YCBN,DYCB(NB),
      & DSIDE(4,NB),DSIDE(2,NB)
1055 FORMAT(9X,I3,4X,4(2X,E13.6)/9X,I3,4X,4(2X,E13.6))
60 CONTINUE
70 CONTINUE
IF(NBN.EQ.NBND) GO TO 80
ERROR=2
GO TO 120
WRITE OUT SLRACE CONDITIONS ON BOUNARY ELEMENTS
80 JC 100 J=1,NYREGN
DO 90 I=1,NXREGN
IF (IELTYP(I,J).LE.0) GO TO 90
NB=IELTYP(I,J)
-1=INTERNAL, 0=SPECIFIED FLOW, 1=SHORELINE, 2=OPEN
WRITE IHRND,IFHRND,JHRND,JPHRND
IF(LPUNC1) WRITE(CATCLT,1060) ISICE(4,NB),ISIDE(2,NB),
      & ISIDE(1,NB),ISIDE(3,NB)
1060 FORMAT(4(2X,I3))
      IF(LPINT1) WRITE(CUT,1065) ISIDE(4,NB),ISIDE(2,NB),
      & ISIDE(1,NB),ISIDE(3,NB)
1065 FORMAT(1X,4(2X,I3))
90 CONTINUE
100 CONTINUE
IF(LPUNC1) ACARDS=3+NXREGN+NYREGN+NEXTHN+3*NBN
GO TO 180
PROCESS ERRCE CCNCITICA
120 IF(DATOUT.EQ.0LT.CF.NET,LPUNC1) GO TO 150
ENDFILE DATCLT
REWIND DATCLT
130 READ(DATOUT,1070,END=140) CCPCY
1070 FORMAT(20A4)
WRITE(OUT,1075) CCPCY
1075 FORMAT(1X,20A4)
GO TO 130
REWIND DATOLT
140 IF(ERROR.EQ.2) GO TO 160
WRITE(OUT,1080) AXI,NEXTFA
1080 FORMAT(////' PREFR2 - CATDEK (DATA OUTPUT) * * * * *'/
      & ' ERROR IN EXTERNAAL ELEPEMNT TALLY FOUND',I4,
      & ' ELEMENTS WHEN SHCULD HAVE',I4)
GO TO 180
160 WRITE(OUT,1085) NBN,NBND
1085 FORMAT(////' PREFR2 - CATDEK (DATA OUTPUT) * * * * *'/
      & ' ERROR IN BCUADARY ELEMENT TALLY, FOUND',I4,
      & ' ELEMENTS WHEN SHCULD HAVE',I4)
180 WRITE(OUT,1090) ACARDS
1090 FORMAT(//' DATDEK-' ,I6, ' CARDS PUNCHED')
RETURN
END

```

```

SUBROUTINE ICRCS(XS,XYS,YS,DYS,XT,DXT,VT,DYT,S,T)

  FIND INTERSECTION POINT OF TWO LINES LYING IN A PLANE
  WITH PARAMETRIC REPRESENTATION  $X(T)=X1+DX1*T$  AND
   $Y(T)=Y1+DY1*T$ 
  THE ALGORITHM IS GENERAL FOR NON-COLLINEAR LINES
  WHEN LINES ARE COLLINEAR A SPECIAL CONVENTION IS USED
  WHICH IS CONSISTANT WITH THE PREPR2 PROGRAM.

  CNS=(XS-XT)*DYT-(YS-VT)*DXT
  CNT=(XT-XS)*DYS-(YT-YS)*DXS
  C1=DYS*DXT
  C2=DYT*DXS
  CD=C1-C2
  CT=ST=(ABS(C1)+ABS(C2))*1.E-6
  IF(ABS(CD).LT.CTEST) GO TO 10
  S=CNS/CD
  T=CN1/CT
  GO TO 90

1  IF(ABS(CNS)+ABS(CNT).GT.CTEST) GO TO 8)

  LINES PARALLEL AND COLINEAR
  DDY=DYS-DYT
  ADY=ABS(DDY)
  DDY=DYS-DYT
  DDY=ABS(DDY)
  ADX=DXS-DXT
  ADX=ABS(ADX)
  IF(ADY.GT.ADX) GO TO 50
  IF(ADY.GT.0.) GO TO 30
  IF(DXS.EQ.0.) GO TO 20
  SECOND LINE HAS ZERO LENGTH
  T=1.
  S=(XT-XS)/DXS
  RETURN

2  FIRST LINE HAS ZERO LENGTH
  S=1.
  IF(DXT.NE.0.) T=(XS-XT)/DXT
  IF(DYT.EQ.0.) T=(YS-VT)/DYT
  RETURN

3  SLOPE MAGNITUDE LESS THAN OR EQUAL UNITY
  IF(DDX+DDY.GT.0.) GO TO 40
  SEGMENTS SIMILARLY DIRECTED
  S=MIN1(((XT-XS)+DXT)/DXS,1.)
  T=MAX1(((XS-XT)+DXS)/DXT,1.)
  GO TO 90

4  SEGMENTS OPPOSITELY DIRECTED
  S=0.
  T=(XT-XS)/DXT+1.
  GO TO 90

5  SLOPE MAGNITUDE GREATER THAN UNITY
  IF(DDX+DDY.GT.0.) GO TO 60
  SEGMENTS SIMILARLY DIRECTED
  S=MIN1(((YT-YS)+DYT)/DYS,1.)
  T=MAX1(((YS-VT)+DYS)/DYT,1.)
  GO TO 90

6  SEGMENTS OPPOSITELY DIRECTED
  S=0.
  T=(YT-YS)/DYS+1.
  GO TO 90

  LINES PARALLEL, NOT COLLINEAR
  S=1.E-6
  T=1.E-6
  GO RETURN
END

```

```

SUBROUTINE SHRAAF(NSHCR,NF,X,Y,NC)
  RETURNS SHORELINE COORDINATES

  DIMENSION NSPTS(10),NSTRT(10),XS(300),YS(300)
  COMMON/INOUT/IN,CLT,OUTOUT
  INTEGER OUT,OUTCLT
  DATA NSPDIM,NXYDIM,NSFTS,NSTRT/10,300,10*0,10*0/
  NSPDIM IS DIMENSION OF ARRAYS NSPTS AND NSTRT
  NXYDIM IS DIMENSION OF ARRAYS XS AND YS

  IF(NC.GT.0) GC TO 50
  IF(NC.EQ.0) GC TO 40

  NC=-1, INITIALIZATION CALL, READ DATA
  READ NUMBER OF DISCONNECTED SHORELINES
  READ (IN,1000) NSHRMX
  1000 FORMAT(I5)
  IF(NSHRMX.LE.NSPDIM) GO TO 10
  WRITE (OUT,1005) NSHRMX,NSPDIM
  1005 FORMAT(//' SHRMX - * * * WARNING * * */1X,14,
  & ' IS TOO MANY SHORELINES, ONLY',13,' SHORELINES',
  & ' ALLOWED BY PRESENT ARRAY DIMENSIONS')
  NSHRMX=NSPDIM

  10 WRITE (OUT,1010) NSHRMX
  1010 FORMAT(///' SHORELINE DATA FOLLOWS.',14,' SHORELINES')

  NCNT=1
  DO 30 N=1,NSHRMX
    READ NUMBER OF POINTS ON SHORELINE N
    READ (IN,1000) NS
    WRITE (OUT,1015) N
  1015 FORMAT(//' SHORELINE ',13/2X,2('NO.',3X,
  & ' X-COORDINATES',3X,' Y-COORDINATES',10X))
    NSPTS(N)=NS
    NSTRT(N)=NCNT-1
    NSTP=NSTRT(N)+NS
    IF(NSTP.LE.NXYDIM) GC TO 15
    WRITE (OUT,1025) NXYDIM
  1025 FORMAT(//' SHRMX - * * * WARNING * * */10X,'TOO MANY'
  & ' SHORELINE POINTS, ONLY',14,' TOTAL POINTS ALLOWED'
  & ' BY PRESENT ARRAY DIMENSIONS'//)
    NSTP=NXYDIM
    NSHRMX=NSHRMX-1
    READ COORDINATES OF POINTS ON SHORELINE
  15 READ (IN,1020) (XS(I),YS(I),I=NCNT,NSTP)
  1020 FORMAT(2F10.0)
    I1=NSTRT(N)
    NSH=NS/2
    NSP=NS-NSH
    I2=I1+NSP
    DO 20 I=1,NSH
      NN=NSP+I
  20 WRITE (OUT,1030) I,XS(I1+I),YS(I1+I),NN,XS(I2+I),YS(I2+I)
  1030 FORMAT(1X,13,2F16.2,10X,13,2F16.2)
    IF(NSP.GT.NSH) WRITE (OUT,1030) NSP,XS(I2),YS(I2)
    IF(NSTP.EQ.NXYDIM) RETURN
  30 NCNT=NSTP+1
    NSHOR=NSHRMX
    RETURN

  RETURN NUMBER OF POINTS ON SHORELINE NSHOR
  40 NP=0
  IF(NSHOR.LE.NSHRMX) NF=NSFTS(NSHOR)
  RETURN

  RETURN COORDINATES OF SPECIFIED POINT ON SHORELINE
  50 IF(NSHOR.GT.NSHRMX) GC TO 60
  IF(NP.GT.NSFTS(NSHOR)) RETURN
  I=NSTRT(NSHOR)+NP
  X=XS(I)
  Y=YS(I)
  RETURN
  60 X=0.
  Y=0.
  RETURN
  END

```


APPENDIX C

**Identification of Some Variables
Used in the Program PREPR2**

Identification of Some Variables Used in the Program PREPR2

AREA -	wetted area of boundary element enclosed by a portion of the shoreline and the element edges.
AREAR -	area of the rectangular element bounded by the grid lines.
BAR -	cumulative wetted area of boundary element.
BXM -	cumulative first moment of the area of an element about the x-axis, = y centroid/area.
BYM -	cumulative first moment of the area of an element about the y-axis, = x centroid/area.
DATOUT -	logical output (punch) unit on which data set is written.
DSIDE -	array containing wetted length of each side of each boundary element, sides ordered counterclockwise with bottom side first.
DXCB -	width along x-axis of wetted area of element as measured through the centroid, same as DXCBND in FAROUT code.
DYCB -	width along y-axis of wetted area of element as measured through the centroid, same as DYCBND in FAROUT code.
I, IC, II -	column index of element.
IADD -	column index increment, used to find column index of element adjacent to any specified side of an element, $IADD(K) = MOD(k/2,2) - MOD((K+3)/2,2)$, $K \geq 0$.
ICC1 -	array of increments to step around sides of a rectangle in counterclockwise direction, $ICC1(K) = 1 + MOD(K,4)$, $K > 0$.
ICC2 -	array of increments to step around sides of a rectangle in counterclockwise direction on steps of two, $ICC2(K) = 1 + MOD(K+1,4)$.
ICW1 -	array of increments to step around sides of a rectangle in a clockwise direction, $ICW1(K) = 1 + MOD(K+2,4)$
IELTYP -	array designating element type, IELTYP = -1 for external element, 0 for internal element and >0 for boundary element. When positive, it specifies the location of element data in boundary element array. Function is similar to array INTEXT in FAROUT but convention is different.

BLANK PAGE

IN - variable designating logical input unit.

ISIDE - array specifying boundary element edge boundary conditions corresponding to DSIDE values. ISIDE = -1 means element borders on internal element, = 0 borders on specified flow regime, = +1 borders on external element, = 2 borders on open flow regime.

IXCB - array containing column index of each boundary element.

J, JC, JJ - row index of element.

JADD - row index increment, $JADD(K+1) = IADD(K)$ (see IADD).

JYCB - array containing row index of each boundary element.

LBND - dimension (number of storage locations) of the boundary arrays.

LPRINT - logical variable controlling printing of output data.

LPUNCH - logical variable controlling punching of output data.

LSEG - array containing list-linked data shoreline entries into boundary elements.

MX - array of column increments for addressing coordinates of corners of rectangular elements.

MY - array of row increments for addressing coordinates of corners of rectangular elements.

NB - index used to reference boundary elements.

NBND - current number of boundary elements.

NBREC - array pointing to entries in array LSEG.

NEXTRN - number of external elements found.

NX1, NXE - number of x-coordinates defining element edges, equal to $NXREGN + 1$.

NXREGN, NX - number of columns of elements.

NY1, NYE - number of y-coordinates defining element edges, equal to $NYREGN + 1$.

NYREGN, NY - number of rows of elements.

OUT - variable defining logical output unit for printing.

TRACE - logical variable controlling the printing of data tracing program flow.

X - array containing x-coordinates of element edges.

XCEN - x-coordinate of element centroid.

Y - array containing y-coordinates of element edges.

YCEN - y-coordinate of element centroid.

APPENDIX D

**Partial Output Listing From PREPR2 Run,
Atlantic Generating Station**

112	17	21170.00	52707.00	36	10000.00	00550.00
113	17	21000.00	52530.00	35	12000.00	07000.00
114	17	22000.00	53000.00	36	12000.00	07000.00
115	18	23070.00	52500.00	37	12000.00	00200.00
116	18	24100.00	51100.00			
117						
118						
119						
120						
121						
122						
123						
124						
125						
126						
127						
128						
129						
130						
131						
132						
133						
134						
135						
136						
137						
138						
139						
140						
141						
142						
143						
144						
145						
146						
147						
148						
149						
150						
151						
152						
153						
154						
155						
156						
157						
158						
159						
160						
161						
162						
163						
164						
165						
166						
167						
168						
169						
170						
171						
172						
173						
174						
175						
176						
177						
178						
179						
180						
181						
182						
183						
184						
185						
186						
187						
188						
189						
190						
191						
192						
193						
194						
195						
196						
197						
198						
199						
200						
201						
202						
203						
204						
205						
206						
207						
208						
209						
210						
211						
212						
213						
214						
215						
216						
217						
218						
219						
220						
221						
222						
223						
224						
225						
226						
227						
228						
229						
230						
231						
232						
233						
234						
235						
236						
237						
238						
239						
240						
241						
242						
243						
244						
245						
246						
247						
248						
249						
250						
251						
252						
253						
254						
255						
256						
257						
258						
259						
260						
261						
262						
263						
264						
265						
266						
267						
268						
269						
270						
271						
272						
273						
274						
275						
276						
277						
278						
279						
280						
281						
282						
283						
284						
285						
286						
287						
288						
289						
290						
291						
292						
293						
294						
295						
296						
297						
298						
299						
300						

[illegible]

```

504 MODEL- TERMINANT NO. 33 TERMINATES IN ICA 2 JCA 27 A.V. 14000.000 46700.000
505 MODEL- TERMINANT NO. 34 TERMINATES IN ICA 2 JCA 27 A.V. 14000.000 46700.000
506 MODEL- TERMINANT NO. 35 TERMINATES IN ICA 2 JCA 27 A.V. 15000.000 47000.000
507 MODEL- TERMINANT NO. 36 TERMINATES IN ICA 2 JCA 27 A.V. 15000.000 47000.000
508
509
510 MODEL- END MARKING AND IMPORTANT SKELETON
511 DISCARD THE OTHER AMPS IN BOUNDARY ELEMENT 1-2 2 27
512
513 MODEL- FIRST BOUNDARY ELEMENT 1-2 2 27 AT SIDE 3 A.V. 15025.000 47500.000
514 AREA = 1.1907E+07 CENTROID A.V. 12510.125 47511.114
515 ISIDE = 1 -1 -1 -1 OSIDE = 5520.300 100.000
516 MODEL- TERMINANT NO. 37 TERMINATES IN ICA 2 JCA 28 A.V. 15030.000 47500.000
517 MODEL- TERMINANT NO. 38 TERMINATES IN ICA 2 JCA 28 A.V. 15030.000 47500.000
518 MODEL- TERMINANT NO. 39 TERMINATES IN ICA 2 JCA 28 A.V. 16000.000 48000.000
519 MODEL- FIRST BOUNDARY ELEMENT 1-2 2 28 AT SIDE 1 A.V. 16000.000 48000.000
520 AREA = 1.1907E+07 CENTROID A.V. 13044.275 48030.010
521 ISIDE = 1 -1 -1 -1 OSIDE = 6430.550 4000.000
522
523 MODEL- SKELETON 4 ENTERS SIDE 1 BY A.V. 15050.000 48000.000
524
525 MODEL- TERMINANT NO. 40 TERMINATES IN ICA 2 JCA 29 A.V. 17500.000 48500.000
526 MODEL- FIRST BOUNDARY ELEMENT 1-2 2 29 AT SIDE 2 A.V. 17500.000 48500.000
527 AREA = 0.5000E+05 CENTROID A.V. 18000.000 48500.000
528 ISIDE = 1 -1 -1 -1 OSIDE = 0.000 500.000
529 MODEL- TERMINANT NO. 41 TERMINATES IN ICA 3 JCA 30 A.V. 19100.000 48500.000
530 MODEL- TERMINANT NO. 42 TERMINATES IN ICA 3 JCA 30 A.V. 19000.000 48700.000
531 MODEL- TERMINANT NO. 43 TERMINATES IN ICA 3 JCA 30 A.V. 19000.000 48700.000
532 MODEL- TERMINANT NO. 44 TERMINATES IN ICA 3 JCA 30 A.V. 19375.000 48000.000
533 MODEL- FIRST BOUNDARY ELEMENT 1-2 3 30 AT SIDE 4 A.V. 17500.000 47500.231
534 AREA = 2.7150E+07 CENTROID A.V. 19050.000 47500.231
535 ISIDE = 1 -1 -1 -1 OSIDE = 4000.000 000.000
536 MODEL- TERMINANT NO. 45 TERMINATES IN ICA 2 JCA 30 A.V. 16750.000 48000.000
537
538
539 MODEL- TERMINANT NO. 46 TERMINATES IN ICA 1 JCA 15 A.V. 1500.000 10000.000
540 MODEL- TERMINANT NO. 47 TERMINATES IN ICA 1 JCA 15 A.V. 1700.000 17100.000
541 MODEL- TERMINANT NO. 48 TERMINATES IN ICA 1 JCA 15 A.V. 200.000 10000.000
542
543 MODEL- END MARKING AND IMPORTANT SKELETON
544 DISCARD THE OTHER AMPS IN BOUNDARY ELEMENT 1-2 1 15
545
546 MODEL- FIRST BOUNDARY ELEMENT 1-2 1 15 AT SIDE 1 A.V. 170.000 10000.000
547 AREA = 1.0000E+05 CENTROID A.V. 610.700 10000.000
548 ISIDE = 1 -1 -1 -1 OSIDE = 4200.000 1000.000
549 MODEL- FIRST BOUNDARY ELEMENT 1-2 1 15 AT SIDE 2 A.V. 210.000 10750.000
550 AREA = 1.0000E+05 CENTROID A.V. 675.000 10750.000
551 ISIDE = 1 -1 -1 -1 OSIDE = 4000.000 1250.000
552 MODEL- FIRST BOUNDARY ELEMENT 1-2 1 15 AT SIDE 3 A.V. 200.000 10000.000
553 MODEL- FIRST BOUNDARY ELEMENT 1-2 1 15 AT SIDE 4 A.V. 0.000 10000.000
554 ISIDE = 1 -1 -1 -1 OSIDE = 0.000 1000.000
555
556 MODEL- SKELETON 4 ENTERS SIDE 1 BY A.V. 1500.000 10000.000
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

[illegible]

ATLANTIC OCEANOGRAPHIC STATIONS, BY DATE																																
IDENTIFICATION OF ELEMENTS AS TO TYPE AND CHARACTER																																
ST	F	001	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
1203																																
1204																																
1205																																
1206																																
1207																																
1208																																
1209																																
1210																																
1211																																
1212																																
1213																																
1214																																
1215																																
1216																																
1217																																
1218																																
1219																																
1220																																
1221																																
1222																																
1223																																
1224																																
1225																																
1226																																
1227																																
1228																																
1229																																
1230																																
1231																																
1232																																
1233																																
1234																																
1235																																
1236																																
1237																																
1238																																
1239																																
1240																																
1241																																
1242																																
1243																																
1244																																
1245																																
1246																																
1247																																
1248																																
1249																																
1250																																
1251																																
1252																																
1253																																
1254																																
1255																																
1256																																
1257																																
1258																																
1259																																
1260																																
1261																																
1262																																
1263																																
1264																																
1265																																
1266																																
1267																																
1268																																
1269																																
1270																																
1271																																
1272																																
1273																																
1274																																
1275																																
1276																																
1277																																
1278																																
1279																																
1280																																
1281																																
1282																																
1283																																
1284																																
1285																																
1286																																
1287																																
1288																																

ARBITRARY GEOMETRIC STATISTICS BY QUANTILE					
	72	76	78	79	80
1367					
1368					
1369					
1370					
1371					
1372					
1373					
1374					
1375					
1376					
1377					
1378					
1379					
1380					
1381					
1382					
1383					
1384					
1385					
1386					
1387					
1388					
1389					
1390					
1391					
1392					
1393					
1394					
1395					
1396					
1397					
1398					
1399					
1400					
1401					
1402					
1403					
1404					
1405					
1406					
1407					
1408					
1409					
1410					
1411					
1412					
1413					
1414					
1415					
1416					
1417					
1418					
1419					
1420					
1421					
1422					
1423					
1424					
1425					
1426					
1427					
1428					
1429					
1430					
1431					
1432					
1433					
1434					
1435					
1436					
1437					
1438					
1439					
1440					
1441					
1442					
1443					
1444					
1445					
1446					
1447					
1448					
1449					
1450					
1451					
1452					
1453					
1454					
1455					
1456					
1457					
1458					
1459					
1460					
1461					
1462					
1463					
1464					
1465					
1466					
1467					
1468					
1469					
1470					
1471					
1472					
1473					
1474					
1475					
1476					
1477					
1478					
1479					
1480					
1481					
1482					
1483					
1484					
1485					
1486					
1487					
1488					
1489					
1490					
1491					
1492					
1493					
1494					
1495					
1496					
1497					
1498					
1499					
1500					
1501					
1502					
1503					
1504					
1505					
1506					
1507					
1508					
1509					
1510					
1511					
1512					
1513					
1514					
1515					
1516					
1517					
1518					
1519					
1520					
1521					
1522					
1523					
1524					
1525					
1526					
1527					
1528					
1529					
1530					
1531					
1532					
1533					
1534					
1535					
1536					
1537					
1538					
1539					
1540					
1541					
1542					
1543					
1544					
1545					
1546					
1547					
1548					
1549					
1550					
1551					
1552					
1553					
1554					
1555					
1556					
1557					
1558					
1559					
1560					
1561					
1562					
1563					
1564					
1565					
1566					
1567					
1568					
1569					
1570					
1571					
1572					
1573					
1574					
1575					
1576					
1577					
1578					
1579					
1580					
1581					
1582					
1583					
1584					
1585					
1586					
1587					
1588					
1589					
1590					
1591					
1592					
1593					
1594					
1595					
1596					
1597					
1598					
1599					
1600					