

AD-A110 275

LITTLE (ARTHUR D) INC CAMBRIDGE MA
DEVELOPMENT OF A HACS USER INTERFACE MODULE. (U)
SEP 81 R G POTTS

F/G 9/2

UNCLASSIFIED

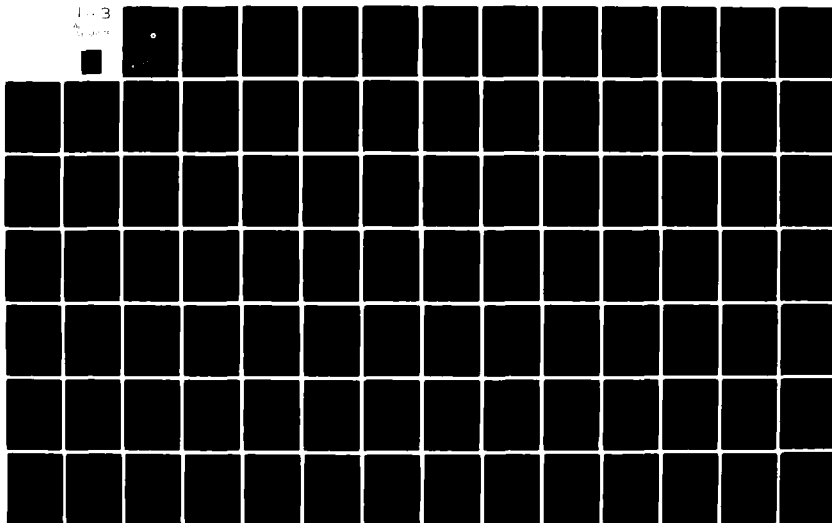
USC6-D-74-81

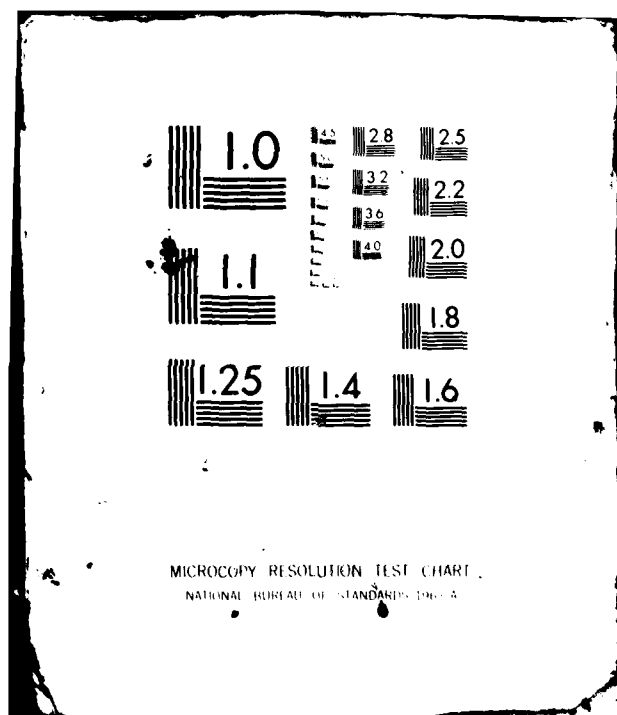
DOT-C6-925331-A
NL

1-3

AL

1-3





Report No. CG-D-74-81

LEVEL #

(12)

DEVELOPMENT OF A HACS USER INTERFACE MODULE

AD A110275



FINAL REPORT

DTIC
SERIALS
FEB 2 1982
A

Document is available to the public through the
National Technical Information Service,
Springfield, Virginia 22161

SEPTEMBER 1981

Prepared for

**DEPARTMENT OF TRANSPORTATION
UNITED STATES COAST GUARD**
Office of Research and Development
Washington, D.C. 20593

FILE COPY

FILE COPY

82 02 01 131

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The contents of this report do not necessarily reflect the official view or policy of the Coast Guard; and they do not constitute a standard, specification, or regulation.

This report, or portions thereof may not be used for advertising or sales promotion purposes. Citation of trade names and manufacturers does not constitute endorsement or approval of such products.

1. Report No. CG-D-74-81	2. Government Accession No. AD-A110 275	3. Recipient's Catalog No.	
4. Title and Subtitle Development Of A HACS User Interface Module		5. Report Date 30 September 1981	
		6. Performing Organization Code	
7. Author(s) Richard G. Potts		8. Performing Organization Report No.	
9. Performing Organization Name and Address Arthur D. Little, Inc. Acorn Park Cambridge, MA 02140		10. Work Unit No. (TRIS) 4156.4	
		11. Contract or Grant No. DOT-CG-925331-A	
12. Sponsoring Agency Name and Address U. S. Coast Guard Office of Research and Development Washington, D. C. 20593		13. Type of Report and Period Covered FINAL REPORT	
		14. Sponsoring Agency Code G-DMT-1/54	
15. Supplementary Notes The U. S. Coast Guards' Research and Development Technical Representative for the work performed herein was Dr. M. C. Parnarouskis			
16. Abstract DEVELOPMENT OF A HACS USER INTERFACE MODULE The Hazard Assessment Computer System (HACS) is one of six major components of the U.S. Coast Guard's Chemical Hazards Response Information System (CHRIS). HACS is a computerized system consisting of chemical spill models and containing all necessary physical and chemical property data to permit hazard assessments to be performed for 900 commonly shipped chemicals. The User Interface Module (UIM) provides for fully interactive operation of HACS with remote access for users to the central computer facility by means of remote terminals. The final report provides documentation of the internal structure of the HACS/UIM as developed for use by the USCG on the Cybernet System of Control Data Corporation. The UIM controls the user interaction with HACS by means of a question and answer dialog; all user responses are interpreted by the UIM without input formatting restrictions using a series of terminal input utility functions. An overview of the internal structure of HACS/UIM is given in terms of both sub-program modules and sequence of operations. Descriptions of all required HACS data files are given; these include a chemical and physical property data file for the 900 hazardous chemicals. Additional descriptive text and detailed input data explanations are stored in external files for retrieval and display under user control. Complete program listings for the HACS/UIM and associated computer programs are included in the report.			
17. Key Words Hazard Assessment, HACS, CHRIS, Hazardous Chemical's		18. Distribution Statement Document available through National Technical Information Service Springfield, Virginia 22161	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 251	22. Price

208350

HAZARD ASSESSMENT
COMPUTER SYSTEM

"DEVELOPMENT OF A HACS USER INTERFACE MODULE"
FINAL REPORT

30 September 1981

Accession For	
NTIS GPOSI	☒
DTIC TAB	☐
Unannounced	☐
Justification	☐
By	
Distribution	☐
Availability	☐
Dist	
<i>A</i>	



Report prepared by:

Richard G. Potts
ARTHUR D. LITTLE, INC.
Acorn Park
Cambridge, Massachusetts 02140

Report prepared for:

DEPARTMENT OF TRANSPORTATION
UNITED STATES COAST GUARD
Washington, D.C. 20593

TABLE OF CONTENTS

Section	Title	Page
1	INTRODUCTION.....	1
2	COMPONENTS OF THE CHEMICAL HAZARD RESPONSE INFORMATION SYSTEM (CHRIS).....	4
2.1	A Condensed Guide to Chemical Hazards (COMDTINST M16465.11).....	4
2.2	Hazardous Chemical Data (COMDTINST M16465.12).....	4
2.3	Hazard Assessment Handbook (COMDTINST M16465.13)....	4
2.4	Hazard Assessment Computer System.....	5
2.5	Response Methods Handbook (COMDTINST M16465.14)....	5
2.6	Data Base For Regional Contingency Plan.....	5
3	OVERVIEW OF INTERNAL PROCESSING.....	6
3.1	Overlay Structure.....	8
3.2	Sequence of Operations.....	9
4	HACS DATA FILE STRUCTURES.....	11
4.1	Chemical Properties File.....	11
4.2	Default/State/Save Files.....	12
4.3	Data Field Explanations (11).....	15
4.4	Scenario Descriptions (12).....	16
4.5	Model Descriptions (13).....	17
5	HACS/UIM PROGRAM LISTINGS.....	18
6	CYBERNET USE.....	181
7	ASSOCIATED PROGRAMS.....	182
7.1	Chemical Property File Manipulation.....	183
7.1.1	Model and Scenario Coding.....	183
7.1.2	Variable to Fixed Length Conversion.....	189
7.1.3	Fixed to Variable Length Conversion.....	192
7.1.4	CDC 3300 Conversion.....	195
7.1.5	Recognition Code/Model Cross-Reference.....	197
7.1.6	Recognition Code/Assessment Path Cross-Reference.....	200
7.1.7	Data Gap Identification.....	203
7.2	Message File Creation.....	208
7.2.1	Field Text (File RGPTXT).....	209
7.2.2	Scenario Text (File RPSTXT).....	216
7.2.3	Model Text (File MTXLOD).....	223
7.3	Message File Display.....	225
7.3.1	Field Text (File RGPMMSG).....	226
7.3.2	Scenario Text (File RPSMSG).....	229

TABLE OF CONTENTS (Continued)

Section	Title	Page
	7.3.3 Model Text (File RGPEDT).....	232
7.4	Utilities.....	233
	7.4.1 Data Compression (File SAVCOD).....	234
	7.4.2 Tab Key (File SAVCNV).....	251

1. INTRODUCTION

The Chemical Hazard Response Information System (CHRIS) is designed to provide timely information essential for proper decision-making by responsible Coast Guard personnel and others during emergencies involving the water transport of hazardous chemicals. A secondary purpose is the provision of certain basic non-emergency-related information to support the Coast Guard in its efforts to achieve improved levels of safety in the bulk shipment of hazardous chemicals.

CHRIS consists of four reference guides or manuals, a regional contingency plan, and a hazard assessment computer system (HACS). The four manuals contain chemical data, hazard assessment methods, and response guides. Regional data are included in the Coastal Regional Contingency Plans. Elements of the Coast Guard Headquarters staff operate the hazard assessment computer system and provide technical assistance on request by field personnel during emergencies. In addition, they are responsible for periodic update and maintenance of CHRIS. A brief description of each major component of CHRIS is provided in Section 2 of this report.

This report documents the internal computer program structure of the Hazard Assessment Computer System (HACS) as it operates on Control Data Corporation's Cybernet Service with the User Interface Module (UIM). A second report, the HACS/UIM Users' Operation Manual, contains additional information which, although not duplicated in this report, is useful for understanding the internal operation of the HACS/UIM. A third report, the HACS Program Reference Manual, contains detailed technical documentation of the earlier batch version of the system as it operated on the CDC 3300 computer at USCG Headquarters; although the batch operation has now been replaced by an interactive mode, much of this earlier documentation gives useful background on the current internal structure of the HACS/UIM.

HACS is comprised of a specific set of hazard assessment models, chemical specific data, and an overall system structure to provide data control and output displays. Two separate computer programs are used to define and produce independent displays of the chemical specific data; the use and capabilities of these programs are described in separate reports.

Also to assist in obtaining the compound recognition code used to reference data for a particular chemical, a separate set of indices have been produced and are given in a separate report. These indices enable a user of HACS to obtain a recognition code for a chemical given either the compound name or a synonym.

In addition to the file of chemical specific data, HACS uses another independent set of data referred to as the default file. Default file data is used by HACS during execution to define the structure of internal HACS processing files. The elements and current data contained in the default file are described in the HACS User Reference Manual. The HACS default file provides the primary control within the UIM for interactive requests for

user input data.

The individual hazard assessment models contained in HACS are basically organized as separate programs, arranged as overlays, which are retrieved and executed under the control of the HACS executive system. Each separate program includes a framework within which data items (environmental conditions, chemical properties, model parameters, user options, etc.) are transferred from HACS data files to the rate model, and results obtained from model computations are stored in HACS files. The program code for the individual assessment models, and associated subroutines, is included in this report, however, the theoretical basis for these models is described in Assessment Models in Support of the Hazard Assessment Handbook published by the U.S. Coast Guard in January 1974. A later report entitled "Development of Additional Hazard Assessment Models" describes new models developed and incorporated into HACS during 1976.

This manual assumes a thorough familiarity with the contents of the HACS User Reference Manual, and uses the principles of operations described in that manual as the basis for technical documentation of the internal program structure.

Most HACS program code is written in Fortran IV, avoiding machine dependent features or coding as much as possible. However, the development of the UIM and associated capabilities for the operation of HACS, and the extended file storage requirements for explanatory text, have required a greater use of specialized coding than was necessary with earlier versions of HACS. The purpose and function of these portions of HACS are described in the detailed program listings; for operation of HACS on other computer equipment, changes in programming syntax will be necessary to preserve the same functions.

In overview, HACS consists of:

- (1) A large operating program for the HACS/UIM,
- (2) A set of procedures used on Cybernet for program execution and supporting operations, and
- (3) A set of additional computer programs which provide supporting functions primarily for maintenance or conversion of data files used by the HACS/UIM.

Sections 3 and 4 of this report contain general information regarding the current structure and operation of the HACS/UIM. Section 5 contains the detailed HACS/UIM computer program listings, each with narratives and identification of all internal variables or parameters. Section 6 gives a brief description of the procedure used on Cybernet to run the HACS/UIM; details of the use of Cybernet services for editing, program compilation and other similar operations required for program development are beyond the scope of this report. Details of the additional supporting computer

programs which were prepared during the UIM project are given in Section 7. This section contains statements of program purpose and function, and complete program listings are included.

The HACS/UIM has evolved through a series of stages in which additional desired features have been identified as experience with the interim versions of the UIM has been gained. Readers of this report are cautioned that the HACS/UIM is a dynamic system, and changes are being made continually in response to new desired features or system capabilities, to new computer operations or use within the Coast Guard, and to requirements which result from other related Coast Guard projects. (For example, the preparation of chemical specific physical property estimates will lead to a significant re-structuring of the HACS physical property interface.) Thus the program listings as contained in this report can only document the system at one particular stage in its development.

2. COMPONENTS OF THE CHEMICAL HAZARD RESPONSE INFORMATION SYSTEM (CHRIS)

2.1 A Condensed Guide to Chemical Hazards (COMDTINST M16465.11)

The Condensed Guide contains information needed to help personnel make the proper response in an emergency situation; it is the only CHRIS handbook that will be carried to the actual scene of an accident. It is intended for use by port security personnel and others who may be the first to arrive at the site of an accidental discharge or fire and who need readily available and easily understood information about the hazardous properties of the chemical involved. It will be used to determine the proper actions that should be taken immediately to safeguard life and property and to prevent contamination of the environment.

COMDTINST M16465.11 briefly describes the chemical and biological hazards of various materials so that personnel at the scene of an accident can assess the danger and consider the appropriate large-scale response. It also lists the on-scene information needed for proper use of the Hazard Assessment Handbook (COMDTINST M16465.13). Selected information on each chemical covered by CHRIS is summarized from the more extensive material in the Hazardous Chemical Data Manual (COMDTINST M16465.12) and is presented on a single page.

2.2 Hazardous Chemical Data (COMDTINST M16465.12)

This manual is the cornerstone of CHRIS. For each substance, it lists the specific chemical, physical, and biological data needed for the preparation and use of the other components of the system. It can also be used after the initial response action, when there is sufficient time to use more detailed information than that found in COMDTINST M16465.11. The first of the six pages devoted to each chemical is a duplicate of the corresponding page in COMDTINST M16465.11.

COMDTINST M16465.12 is intended for use primarily by the On-Scene Coordinator (OSC) and by Regional and National Response Centers for devising, evaluating, and carrying out response plans.

Much of the quantitative information found in COMDTINST M16465.12 is needed for the hazard assessment calculations described in the Hazard Assessment Handbook (COMDTINST M16465.13). COMDTINST M16465.12 contains the so-called Hazard Assessment Code, which directs the user of COMDTINST M16465.13 to the appropriate calculation procedures of hazard assessment. COMDTINST M16465.12 also suggests general responses to an accidental discharge, which summarize the detailed information given in the Response Methods Handbook (COMDTINST M16465.14).

2.3 Hazard Assessment Handbook (COMDTINST M16465.13)

The Hazard Assessment Handbook describes procedures to be used for estimating the quantity of a hazardous chemical that may be released

accidentally during shipment. It also describes how to estimate its concentration in air and in water as a function of time and distance from the discharge. Methods for predicting the resulting toxicity, fire, and explosion effects are also described. The calculations use data from COMDTINST M16465.12.

2.4 Hazard Assessment Computer System

The Hazard Assessment Computer System (HACS) is a computerized version of COMDTINST M16465.13. It permits trained headquarters specialists to obtain very detailed hazard evaluations quickly, when requested by OSC personnel.

In addition to computer models for hazard assessment computations, HACS includes several related computer based systems such as the physical property update and data retrieval programs, and the chemical compound name and synonym cross-reference programs. Although HACS and COMDTINST M16465.12 are based on the same original body of data, differences may occasionally arise because HACS is more readily updated than the printed manual.

2.5 Response Methods Handbook (COMDTINST M16465.14)

The Response Methods Handbook is written specifically for Coast Guard OSC personnel who have had some training or experience in hazard and pollution response. The handbook describes cautionary and corrective response methods for reducing and eliminating hazards that result from chemical discharge.

Although several types of response are suggested in COMDTINST M16465.12, the specific response, to be chosen from among those described in COMDTINST M16465.14, should be determined by the results of the hazard assessment procedures in COMDTINST M16465.13.

2.6 Data Base for Regional Contingency Plan

The Coast Guard's Regional Contingency Plans, although not considered a part of CHRIS, are an important adjunct to the system. Each Regional Contingency Plan contains a section (Annex XX) that presents data on a specific region, sub-region, or locale. These data, which are intended for use by OSC personnel, include such information as the following:

- An inventory of physical resources and strike forces;
- Vulnerable or exposed resources (critical water-use areas);
- Potential pollution sources;
- Geographic and environmental features;
- Cooperating organizations; and
- Recognized experts with identified skills.

3. OVERVIEW OF INTERNAL PROCESSING

The HACS/UIM is composed of a number of different program components; these include:

- Executive system - to control the overall sequence of operations, to provide utility functions for assessment model use, and to control the method of assessment model operation with the UIM (i.e., user input and model summary modes).
- Assessment models - a series of separate computer programs retrieving required data from the HACS state file, performing the indicated assessment computations, storing computed values in the state file for subsequent use and generating displays of computed results at the user terminal.
- Overlay control - used by the HACS executive system to select and execute the appropriate portion of the computer program, segmented into overlays, for each step required in an assessment computation.
- Input sequence control - to control the initialization of the HACS state file, user specification of the chemical recognition code and hazard assessment path code, and user selection of output display units.
- Property data processor - to retrieve requested data from the property file, perform unit conversions to internal HACS CGS units, to compute values of functions of temperature, and to transfer chemical specific property data to the HACS state file.
- Terminal input processing - a series of related functions to perform the reading of user entries typed at a terminal in a free format mode, and either transfer the resulting values to other portions of HACS or issue diagnostic messages to the user terminal and process corrections.
- State file interface - a series of routines which control the transfer of data values between the HACS state file and the hazard assessment models. For the UIM, these routines perform additional functions to enable an assessment model, when accessing the state file to obtain a data value not yet entered, to initiate an interactive request to the user terminal to obtain the value. Similarly, the interface also provides for the summary of model input values prior to the execution of the model.
- Message processing - to enable HACS to access external files of descriptive explanations for individual data items, hazard assessment models and hazard assessment scenarios.

- Code interpretation - a series of generalized routines used to translate packed codes from external file structures to forms for internal HACS use. These routines were developed to permit compaction of HACS' external file storage requirements.
- Printer plotting - routines provide a generalized utility producing graphic displays at the user terminal. All scaling functions are performed automatically, and are based on the magnitude of the data points displayed to obtain even or uniform axis labeling. The off-line plotting capability formerly contained in HACS Overlay 2 has been obsolete for some time, and has been deleted from the UIM version.

These components utilize a number of both internal and external files in their operations:

- Program file - a permanent disk file containing pre-compiled HACS program code in overlay and segment structure. The HACS executive system automatically accesses this file to load portions of the HACS program code into computer memory for execution as required.
- State file - an internal program file constructed by HACS during an assessment run; provides data base storage for all user input, property, default and computed data items utilizing a data quality priority structure.
- Default file - a permanent disk file defining the structure of the HACS state file, and containing estimated values for HACS data items to be used only in the absence of any other value.
- Chemical properties file - a magnetic tape file containing predefined physical property data for 900 hazardous chemical substances.
- Save file - an internal program file containing a copy of a HACS state file after completion of user input operations. Permits HACS re-runs requiring only new input values to be used.
- Message files - three external files which contain coded or uncoded variable length text descriptions of HACS data fields, assessment models and assessment scenarios.
- User terminal - although not actually a file in the usual sense, the user terminal is the source of input data requested interactively by the UIM during a hazard assessment computation.

Further descriptions of the functions of these files is contained in Section 2 of the HACS/UIM Users' Operation Manual; additional detailed information regarding file structure and other characteristics is given in Section 4 of this report.

3.1 Overlay Structure

Due to the amount of program code contained in HACS, and memory limitations on the CDC 3300 computer system, the batch version of HACS originally installed at USCG Headquarters consisted of a series of program overlays that are loaded into memory and executed as needed under the control of the HACS executive system. The overlay structure was transferred when HACS was installed on Cybernet, and remains in the UIM version. An over-riding consideration in the design of the overlay structure was the anticipation of future modifications, enhancements or even replacements that might develop as a result of advances in hazard assessment technology. Also the fact that HACS is constructed to proceed along any user specified path of the hazard assessment event chart led to establishing individual rate models within separate overlays.

Overlay capabilities are defined for 3 levels. The main or root overlay, 0, is resident at all times and uses a system utility to load and execute any one of a multiple number of second level overlays. Each of these may, in turn, load and execute any one of a multiple number of third level overlays which are referred to as segments. Each overlay or segment is executed as a separate program. Subroutines resident in a higher level overlay may be referenced. On completion of the execution of the separate overlay or segment program, control returns to the higher level overlay immediately following the location at which control was transferred to the completed overlay or segment.

Overlay 0 contains the program and subroutines comprising the HACS executive system and is resident during all stages of a HACS run. In addition, overlay 0 contains a number of utility routines and library functions used by the rate models, system I/O routines and the overlay and segment control functions. Overlay 1 contains both the input sequence control and property data processors; overlay 2 contained the off-line plot generator and has been deleted.

Assessment rate models are contained in the remaining overlays, 3 to 8. Overlays 3 and 6 contain only a single rate model each. The first part of each of these overlays is coded as a main program which performs the functions of retrieving model input from the HACS state file, calling the appropriate rate model subroutines to perform the indicated assessment computations, then storing the results of these computations in the HACS state file.

Overlays 4, 5 and 7 each contain a multiple number of related, or independent rate models. For each of these, a separate main program exists to interface with the HACS executive system to select the appropriate rate model for execution. Then the first part of each individual model, coded as a subroutine, performs the functions of HACS data base I/O and executes the particular sequence of assessment computations.

Overlay 8 is a small control program, with resident multi-use routines for assessment computations, used to branch to one of six segment level

overlays. Each of segments 1, 3, 4, 5 and 6 are organized in a manner similar to overlays 3 and 6. That is, a separate program provides HACS data base I/O, then subroutines in the segment are called to perform the particular assessment computations. Note that these subroutines may in turn call multi-use subroutines in overlays 8 and/or 0. Segment 2 contains a separate main program to select between models K and P. Each of these models is then controlled by a subroutine which provides separate HACS data base control.

Internal HACS data files are resident in overlay 0.

3.2 Sequence of Operations

The basic HACS functions are to process user inputs to select available options or to enter data values, execute rate models, and display the results of these computations at the user terminal. These operations are sequenced internally by the executive system resident in overlay 0 which first executes overlay 1 to initiate the user input operation for chemical and hazard assessment path selection. When the user entry of the chemical recognition code has been read, HACS automatically accesses the external physical properties file and retrieves the corresponding data record. The chemical property values are uncoded, converted and stored in the HACS state file for later use. Also, at this stage, the codes for appropriate models and scenarios for the chemical are retrieved from this file, and used to control optional displays which can be requested by the user. During the input sequence which occurs in overlay 1, the HACS state file is initialized with default values, and on return to overlay 0 the state file contains only default and chemical property values.

Next, the HACS/UIM uses the user specified list of hazard assessment model codes and proceeds to execute each model by calling the appropriate overlay. The HACS executive initializes a flag for each model to a value of zero; after each execution the flag is set to 1 so that the first execution of each rate model can be identified. As each model is executed a series of subroutine calls are processed to obtain the necessary model input values from the HACS state file. In the state file interface portions of the system, if the value in the state file is a default value and if the model is being executed for the first time, the HACS/UIM does not use the value in the state file but instead issues a prompt at the terminal to obtain a value from the user. User responses (input of a value, query for the current value, or request for description) are processed, and this procedure is repeated for each data item until all model inputs have been obtained. The HACS/UIM then prints a summary of the model input values (by re-executing the model itself a second time), and provides an opportunity for the user to change any previously entered value. If any changes are made, the model input summary is repeated before the hazard assessment computation is done. Following the computation and display of results, the HACS executive processes an optional user request to re-run the model. In this case the user prompts for data input are suppressed, and the model input summary is displayed.

Each hazard assessment model is similarly executed until all models requested by the user have been run. The HACS/UIM then processes a user option to obtain displays of selected values from the HACS state file and the basic run is completed. In overlay 0 and then overlay 1, further user input responses determine whether a new run is initiated or further operation of the program is terminated.

The HACS/UIM Users' Operation Manual contains additional summary discussion of the processing steps of a HACS/UIM run, and a complete description of the terminal displays produced during a typical run.

4. HACS DATA FILE STRUCTURES

HACS utilizes several external and internal files for input data manipulation and storage during operation, in addition to fixed internal data items used for unit label interpretation, output reporting, etc. These files are identified and described below in detail where appropriate or by reference to other documentation.

4.1 Chemical Properties File

The chemical properties file is an external file of physical property data which may be accessed on either magnetic tape or disk; the file currently contains properties for 900 compounds, although additional compounds are expected to be added in the near future. Due to the length of this file, the original version of HACS utilized binary rather than source data formats. This machine dependent format has been retained throughout all later versions of HACS and requires separate translation programs to move copies of the property file from machine to machine.

A detailed description of the original file structure is given in the report "HACS Physical Property File Update and Maintenance - User and Technical Program Documentation."

For use with the UIM, two significant changes have been made to the property file format. First, the hazard assessment model codes originally entered on the file were used to create chemical specific codes for both models and scenarios, and these codes are now contained on the file. Second, a substantial portion of the data elements contained in the file are missing (i.e., not available or not pertinent), and a coding scheme has been used to further compact the file.

The version of the UIM described in this report uses a property file consisting of a header record followed by 900 data records, one per chemical. The header record format has not been changed. Each data record however is now treated as a variable length record, having a maximum length of 84 words. A third record structure, consisting of variable length logical records packed into fixed length physical record blocks, has been developed for use on machines not having the required variable length record facilities (see Section 7).

The variable length record format used by the current version of the UIM operating on Cybernet is described below:

<u>Record Element</u>	<u>Description</u>
1	Chemical recognition code, three letter code in integer word.
2 - 6	Chemical name, up to 40 characters in length stored as 8 characters per word in integer format.

<u>Record Element</u>	<u>Description</u>
7	Hazard assessment model codes applicable to particular chemical, stored as single bit settings in single integer word; uses 29 positions in word. Codes are 0 = model not applicable, 1 = model applicable.
8	Scenario codes, stored as single bit settings to correspond to internal HACS data list of scenario codes (e.g., bit 5 set to 1 indicates HACS scenario 5 is applicable).
9 - 13	Property value status codes, stored as 2 bits per code, 15 codes per word. Codes are 0 = missing, 2 = estimate, and 3 = exact. If code for item I is given as 0, then that item does not appear in following data.
14 - 84	Space for up to 71 data values for chemical, starting with molecular weight. All missing values are removed, so that positions 14, 15, etc. contain only actual data values. Excess space remaining, if any, at end of record is truncated giving variable length records (length is determined by number of data items actually present).

4.2 Default/State/Save Files

The HACS default file is a 2489 word external disk file written in binary mode from internal array storage. The HACS state and save files exist internally in HACS common areas and are simply arrays for data storage.

The structure and data organization of these three files are identical, and contain data values for HACS fields at different stages of processing. The default file is used to initialize the HACS state file. Field data values are stored in the HACS state file according to the attributes or characteristics of the field, and these characteristics are also defined by the default file.

The internal representation of the HACS state and save files are defined by the arrays stored in the common area /BASE/; refer to the listing of program HACS. Using the elements of the HACS state file for reference, the structure of each of these 2489 word files is as follows:

<u>Identification</u>	<u>Length (Words)</u>	<u>Description</u>
MSG(10)	10	80-character file label
MNF/MNI	2	Maximum file size allocations MNF = maximum number of real data fields MNI = maximum number of integer data fields
NF/NI	2	Actual file size utilized NF = number of real data fields NI = number of integer data fields
LIST(275,6)	1650	Contains six integer words of control data for each of up to a maximum of 275 data fields (see below).
FVAL(225,3)	675	Contains a value, minimum limit and maximum limit for up to 225 real data fields
IVAL(50,3)	150	Contains a value, minimum limit and maximum limit for up to 50 integer data fields

2489 = Total file length.

Data storage and retrieval operations from these files are governed by the contents of the array LIST(275,6), referred to as the field definition table. The first entry of this table is reserved for the chemical recognition code. For any field I, the entries in the table are:

LIST(I,1) = Field number

LIST(I,2) = Coded field specification which can be written in BCD form as C4-C3-C2-C1 where the characters C1 to C4 give the field specifications as follows:

IVAR = C4, specifies the field value data type as integer (0) or real (1)

ITYP = C3-C2, specifies the type of quantity governing input/output conversions of the field value as a two digit integer index in the range 01 to MTYP

ISRC = C1, specifies the source or priority code associated with the current value of the field, coded as:

- 0 missing
- 1 default
- 2 estimated property
- 3 exact property
- 4 computed value
- 5 user value
- 6 system value

LIST(I,3) } = Up to twelve character field name used for
LIST(I,4) } output displays
LIST(I,5) }

LIST(I,6) = Index to entry in field data value arrays.
Gives index to array IVAL if code C4 (above) is 0, or to array FVAL if code C4 is 1.

During a normal hazard assessment run, the only file items that are changed as a result of user and/or computed assessment operations are MSG, the source code C1 of the field definition table, and data values stored in array positions FVAL(I,1) or IVAL(I,1). All other elements of the state and save files are obtained initially from the default file and cannot be changed during any assessment run.

The various save and recall operations performed by HACS to enter data into, or read data from, these files are controlled by the field definition table, requiring the coding and uncoding of LIST(I,2) as defined below.

- (a) Define field specification - given pre-determined values of IVAR, ITYP and ISRC for field I, the field specification array element is coded as:

$$\text{LIST(I,2)} = 1000 \cdot \text{IVAR} + 10 \cdot \text{ITYP} + \text{ISRC}$$

which can also be written as:

$$\text{LIST(I,2)} = \text{ISRC} + 10 \cdot (\text{ITYP} + 100 \cdot \text{IVAR})$$

Also, if the index stored in the last entry of the table is J = LIST(I,6), then the field values for field I are actually stored in:

$$\text{if IVAR} = 1 \quad \left\{ \begin{array}{ll} \text{FVAL(J,1)} & = \text{value} \\ \text{FVAL(J,2)} & = \text{minimum limit} \\ \text{FVAL(J,3)} & = \text{maximum limit} \end{array} \right.$$

or, if $IVAR = 0$ $\left\{ \begin{array}{l} IVAL(J,1) = \text{value} \\ IVAL(J,2) = \text{minimum limit} \\ IVAL(J,3) = \text{maximum limit} \end{array} \right.$

- (b) Update source code - to change the source code specification for the value of field I, from OLD to NEW, the field specification array element is manipulated as follows:

$$LIST(I,2) = LIST(I,2) - OLD + NEW$$

where both OLD and NEW are integer variables.

- (c) Read field specifications - given an existing field number, the table element $LIST(I,1)$ is searched until a match is found, and the field specification array element is uncoded by the following sequence:

```
IVAR = LIST(I,2)/1000
ISRC = 1000*IVAR
ITYP = (LIST(I,2)-ISRC)/10
ISRC = LIST(I,2)-10*ITYP-ISRC
```

where ISRC is also used for intermediate storage of the product $1000*IVAR$.

All HACS database operations are controlled by the routines IRCL, ISV, FRCL and FSV which are used to recall or save integer or real field values. Calls to these routines are issued by the HACS assessment rate models and define as a literal in the calling argument list the field number to be saved or recalled.

Thus the definitions of field numbers and field data types (real, integer) in the default file are not arbitrary and must correspond exactly to the definitions established by the HACS program. The field type indicator, elements C3 and C2 of $LIST(I,2)$, govern unit labeling and conversions and are not generally arbitrarily defined.

4.3 Data Field Explanations (11)

An external file, unit 11, is used to store text descriptions of each of the data item fields defined in the HACS state file. Individual field descriptions are displayed at the user terminal either during interactive user input, or as part of the final run summary.

The user reference to these messages is controlled either by the UIM, or, during changes to the model input summary, by the user entry of the HACS data item field number. These field numbers are the four digit integer reference numbers (1000 series, 2000 series, 3000 series and 4000 series) defined by the HACS default file and stored in the HACS state file. Within the state file, the data fields are stored sequentially, and the present

version of HACS contains 256 data items. Given a data item reference by field number, the HACS/UIM uses the internal state file for reference and obtains a message index number (1 to 256) from the sequential position of the data item in the state file. Records contained in the data field explanation file are then directly accessed by keying on the message sequence number.

Each record in the file may contain from 3 to 690 characters of data, and the first word of a record may contain a code for different types of messages. The minimum record length of 3 words is a restriction required for Cybernet processing (shorter records are padded with blank fill). Messages containing less than 690 characters of text are written as variable length records.

The file contains four types of messages:

- (1) Uncoded records contain text which is displayed at the user terminal.
- (2) Type 1 records contain only a code '1' followed by blanks. This code is automatically translated by HACS/UIM to a standard message referencing the user manual.
- (3) Type 2 records contain only a code '2' followed by blanks. This code is automatically translated by HACS/UIM to a standard message referencing CHRIS Manual II.
- (4) Type 3 records are a combination of uncoded records and type 2 records. They contain the code 3 followed by variable length text. In HACS, these messages provide for additional explanation or clarification of particular chemical property data items. They are processed by first displaying the message from the external file, followed by the standard type 2 CHRIS Manual II reference.

The message file is created using separate programs described in Section 7 of this report which insert Fortran format codes between each line of a message. When read by the HACS/UIM, appropriate opening and closing format characters are appended to the message text, and the messages are displayed using variable format output: WRITE(6,TEXT).

4.4 Scenario Descriptions (12)

An external file, unit 12, is used to store text descriptions which are used to produce the scenario display selected by user option. The file contains a display header message, scenario descriptions and two display trailer messages.

The file contains 31 variable length, uncoded, text messages, each containing up to 690 characters of data. Messages 1 to 28 give

descriptions of each of the 28 different hazard assessment scenarios, and the messages correspond in sequence to an internal list of scenario codes contained in the HACS/UIM. Message 29 is the display header, and messages 30 and 31 give the display trailer.

The message file is produced by a variation of the program used to create the field text message file (refer to Section 7).

The records contained on the file consist of character strings giving individual lines of the message, each string separated by appropriate Fortran format control characters. These control characters are automatically inserted by the message file build programs (Section 7). When read by the HACS/UIM, appropriate opening and closing format characters are appended to the message text, and the messages are displayed using variable format output: WRITE(6,TXT).

4.5 Model Descriptions (13)

An external file, unit 13, is used to store text descriptions which are used to produce model explanations selected by user option. The file contains 29 messages, one for each model, and the message numbers correspond in sequence to an internal model code list contained in the HACS/UIM.

Each message is variable length, uncoded, and contains up to 1900 characters of data. The message file is produced by a variation of the program used to create the field text message file (refer to Section 7).

The records contained on the file consist of character strings giving individual lines of the message, each string separated by appropriate Fortran format control characters. These control characters are automatically inserted by the message file build programs. When read by the HACS/UIM, appropriate opening and closing format characters are appended to the message text, and the messages are displayed using variable format output: WRITE(6,TXT).

5. HACS/UIM PROGRAM LISTINGS

This section gives complete listings of all programs and subroutines comprising the HACS/UIM installed on CDC's Cybernet as of 20 April 1981, and may be subject to change based on the results of subsequent work.

The listings are given in the sequence in which the HACS overlay program files were established; programs and routines within overlay 0 are followed by programs within overlay 1 and so forth.

Generally, the listing of each routine provides a description of the overall function and method of operation of the routine, definitions of variables used, and commented processing sequences. File and data element structures are defined where primary references occur. In addition, the main program of the base overlay, PROGRAM HACS, and subroutine PROP of overlay 1 contain complete definitions of all common variables used.

OVERLAY(UIMABS,0,0)
 PROGRAM HACS(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE9,TAPE10,
 1TAPE60=INPUT,TAPE61=OUTPUT,TAPE11,TAPE12,TAPE13)

PROGRAM HACS PROVIDES THE OVERALL CONTROLLING FRAMEWORK
 EXECUTING THE HACS INPUT DATA PROCESSOR, SELECTED HAZARD
 ASSESSMENT RATE MODELS AND THE OFF-LINE PLOTTED OUTPUT POST-
 PROCESSOR. THE CONTROLLING LOOP CYCLES UNTIL EITHER A FATAL
 ERROR IS DETECTED IN THE INPUT DATA, OR AN END OF FILE IS
 REACHED. ALL OVERLAYS ARE RESTORED AS REQUIRED BY CALLS TO

SUBROUTINE OVLDD. ERROR EXIT MODE IS TURNED ON BY COMPASS
 ROUTINE *ERR*. THIS ROUTINE WILL TRAP ALL EXECUTION ERRORS
 TO A SUBROUTINE FCHCK WHERE A MESSAGE WILL BE OUTPUT
 INDICATING THE ERROR AND THE OVERLAY IN WHICH IT OCCURRED.
 EXECUTION IS THEN TERMINATED.

FBLNK = DATA WORD SET TO ALL BLANKS (A8) USED TO INITIALIZE
 THE OUTPUT PAGE TITLE

I = DUMMY FORTRAN INDEX

COMMON VARIABLES USED - ANG,DATE,E0F,IFRST,LNCT,LP,MODEL,MODNO,
 NOFF,NOP,NPG,OVLST,PLTYP,STCON,SVCON,
 TITLE

SUBROUTINES REQUIRED - DYYR,FCHCK,LSTFL,OVLDD,PAGER,TRACE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
 35/309A ACORN PARK,
 CAMBRIDGE, MASS., 02140
 TEL. 617-864-5770 EXT. 2813

DATE - 19 FEBRUARY 1976

COMMON/OVER/NOV,SEG

NOV = OVERLAY NUMBER CORRESPONDING TO ASSESSMENT RATE MODEL
 SEG = SECONDARY OVERLAY NUMBER LOADED IN CORE

COMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
 1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
 INTEGER UPTH
 REAL MSG
 DIMENSION STATE(2489)
 EQUIVALENCE (STATE(1),MSG(1))

SAVE = HACS SAVE FILE, CONTAINS SAVED DATA BASE IN SAME
 FILE STRUCTURE AS HACS STATE FILE
 UPTH = CONTAINS CURRENT VALUES OF PATH CODES AS READ FROM
 USER INPUT AND STORED LEFT JUSTIFIED IN UP TO 15
 INTEGER WORDS (A1 FORMAT)
 STATE = HACS STATE FILE CONTAINING ELEMENTS AND STRUCTURED
 AS DEFINED BY THE REMAINDER OF THIS SECTION -
 MSG = A 10 WORD REAL ARRAY GIVING AN 80 CHARACTER HACS
 FILE LABEL. CONTENTS ARE COPIED FROM THE USER
 INPUT TITLE CARD. FOR CREATING AND UPDATING HACS
 DEFAULT FILES, THIS TEXT SHOULD INCLUDE THE NAME(S)
 OF THE AUTHOR OF THE FILE AND DATE OF PREPARATION
 MNF = MAXIMUM ALLOWABLE NUMBER OF REAL FIELD VALUES,
 EQUIVALENT TO DIMENSION OF ARRAY FVAL
 MNI = MAXIMUM ALLOWABLE NUMBER OF INTEGER FIELD VALUES,
 EQUIVALENT TO DIMENSION OF ARRAY IVAL
 NF = CURRENT NUMBER OF REAL FIELD VALUES ACTUALLY STORED
 STORED IN FILE ARRAY FVAL
 NI = CURRENT NUMBER OF INTEGER FIELD VALUES ACTUALLY
 STORED IN FILE ARRAY IVAL
 LIST = FIELD DEFINITION TABLE DIMENSIONED AS LIST(MFLD,6)
 WHERE MFLD=MNI+MNF. NUMBER OF DEFINITIONS ACTUALLY

STORED IS GIVEN BY $NFLD=NI+NF$. FIRST FIELD
DEFINITION IS RESERVED FOR CHEMICAL RECOGNITION
CODE - SEE NOTE BELOW. FOR ANY FIELD I, THE
ELEMENTS OF LIST ARE DEFINED AS FOLLOWS -

LIST(I,1) = FIELD NUMBER
LIST(I,2) = CODED FIELD SPECIFICATION STORED IN A
FOUR DIGIT INTEGER FORMAT C4-C3-C2-C1
WHERE -
C4 = INDICATOR FOR INTEGER (0)
OR REAL (1) FIELD VALUE
C3-C2 = TWO DIGIT INTEGER TYPE
CODE IN RANGE 01 TO
MTYP DEFINING TYPE OF
PHYSICAL QUANTITY,
ALSO USED AS INDEX
CONTROLLING UNITS
CONVERSION
C1 = FIELD VALUE SOURCE CODE
0 = MISSING
1 = DEFAULT
2 = ESTIMATED PROPERTY
3 = EXACT PROPERTY
4 = COMPUTED VALUE
5 = USER VALUE
6 = SYSTEM VALUE
LIST(I,3) = ONE TO TWELVE CHARACTER FIELD NAME
LIST(I,4) = USED FOR OUTPUT DISPLAYS, AND
LIST(I,5) = STORED IN WORDS 3,4 AND 5 OF LIST
LIST(I,6) = INTEGER POINTER, INDEX, TO ARRAYS FOR
ACTUAL FIELD VALUE, AND MINIMUM AND
MAXIMUM VALUE. POINTS TO ARRAY IVAL
IF CODE C4 (ABOVE) IS 0, OR TO ARRAY
FVAL IF CODE C4 IS 1.
IVAL(J,1) = VALUE OF INTEGER FIELD, LINKED TO FIELD
NUMBER IN LIST(I,1) BY INDEX IN
LIST(I,6), AND CODE C4 IN LIST(I,2)
IVAL(J,2) = MINIMUM VALUE OF INTEGER FIELD
IVAL(J,3) = MAXIMUM VALUE OF INTEGER FIELD
FVAL(J,1) = VALUE OF REAL FIELD, SEE IVAL(J,1)
FVAL(J,2) = MINIMUM VALUE OF REAL FIELD
FVAL(J,3) = MAXIMUM VALUE OF REAL FIELD

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP

PLTYP = INTEGER CODE SET BY HACS RATE MODEL TO SELECT LOGIC
FOR PRODUCING OFF-LINE PLOT TAPE FILE
XBX = ARRAY USED TO PASS OFF-LINE PLOT DATA FROM RATE
MODELS TO PLOT LOGIC. DATA IS AVAILABLE IN XRX
ONLY IMMEDIATELY FOLLOWING EXECUTION OF RATE
MODEL AND IS OVERWRITTEN BY EXECUTION OF NEXT
RATE MODEL. EXCESS SPACE IN XBX IS ALSO USED
FOR STORAGE OF ADDITIONAL DATA ARRAYS USED BY
RATE MODELS.

COMMON/CNTRL/EOFF,ICD,IDFLT,LBL(4),LSTCN(3,3),MODEL(15),NOP.
1 INTEGER STCON,SVCON
REAL EOFF,STCON,SVCON
LBL

EOFF = INDICATOR SET TO -1 IF A SET OF USER DATA CARDS HAS
BEEN TERMINATED BY AN END OF FILE, 0 OTHERWISE.
ICD = CHEMICAL RECOGNITION CODE (A4) READ AS USER DATA
IDFLT = FORTRAN UNIT NUMBER FOR EXTERNAL STORAGE OF HACS
DEFAULT DATA FILE MANIPULATED BY SUBROUTINE ACCESS
LBL = FOUR WORD ARRAY OF LABELS DESCRIBING TYPE OF HACS
STATE OR SAVE FILES (1=EMPTY, 2=DEFAULT, 3=USER,

4=COMPUTED)
 LSTCN = ARRAY USED TO STORE VALIDATED FILE DISPLAY OPTIONS
 BY REFERENCE NUMBER FOR USE IN SUBROUTINE LSTFL
 MODEL = ARRAY OF INTEGER RATE MODEL INDICES CORRESPONDING
 TO USER SPECIFIED PATH CODE LETTERS
 NOP = HACS OPERATION CONTROL VARIABLE DETERMINED FROM USER
 INPUT AS - 1 = RUN
 2 = RE-RUN
 3 = CONTINUE
 4 = LOAD DEFAULT
 5 = UPDATE DEFAULT
 STCON = INTEGER CONTROL FOR TYPE OF VALUES IN HACS STATE
 FILE (REFER TO DEFINITION OF ARRAY LBL)
 SVCON = INTEGER CONTROL FOR TYPE OF VALUES IN HACS SAVE
 FILE (REFER TO DEFINITION OF ARRAY LBL)

COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)

CONV = ARRAY OF CONVERSION FACTORS STORED AS CONV(JSYS,ITYP)
 FOR EACH DEFINED TYPE (ITYP) OF PHYSICAL QUANTITY.
 THE SYSTEM INDEX JSYS = 1, 2, 3 CORRESPONDS TO
 UNIT LABEL INDICES ISYS = 2, 3, 4 SINCE CONVERSION
 FACTORS ARE NOT STORED FOR INTERNAL HACS UNITS.
 CONVERSION FACTORS ARE APPLIED AS FOLLOWS -
 (INTERNAL VALUE) = (INPUT VALUE) * CONV
 EXCEPT FOR TEMPERATURE CONVERSIONS WHICH ARE CON-
 TROLLED BY BRANCHING ON IITYP.
 MSYS = MAXIMUM NUMBER OF DEFINED SYSTEMS OF UNITS
 MTYP = MAXIMUM NUMBER OF TYPES OF PHYSICAL QUANTITIES
 UNIT = ARRAY OF FIELD VALUE UNIT LABELS (AB) STORED AS
 UNIT(ISYS,ITYP) FOR EACH DEFINED TYPE (ITYP) OF
 PHYSICAL QUANTITY AND SYSTEM OF UNITS (ISYS). UNIT
 LABELS AND CONVERSION FACTORS ARE DEFINED FOR BOTH
 REAL AND INTEGER FIELDS, HOWEVER, NUMERIC
 CONVERSIONS ARE APPLIED ONLY TO REAL FIELDS. CON-
 VERSIONS FOR ANY TYPE QUANTITY MAY BE SUPPRESSED
 BY LEAVING THE APPROPRIATE DATA WORD IN UNIT BLANK.
 ALL UNITS HOWEVER MUST BE GIVEN FOR SYSTEM 1 WHICH
 SPECIFIES INTERNAL UNITS. ON INPUT, DATA READ WITH
 A BLANK UNIT FIELD IS ASSUMED TO BE IN INTERNAL
 UNITS. ON OUTPUT, VALUES ARE DISPLAYED IN ALL
 DIFFERENT NON-BLANK UNITS SPECIFIED.

COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)

DTE = DATE OF PROGRAM EXECUTION DETERMINED AT RUN TIME
 LNCT = COUNT OF LINES PRINTED ON CURRENT LINE PRINTER PAGE
 LNPG = MAXIMUM NUMBER OF LINES PER LINE PRINTER PAGE
 LP = FORTRAN UNIT NUMBER FOR LINE PRINTER
 NPG = LINE PRINTER PAGE NUMBER
 TITLE = 80 CHARACTER USER INPUT RUN TITLE DISPLAYED AT THE
 TOP OF EACH OUTPUT PAGE.

COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP

ICVSL = OUTPUT UNIT SELECTION OPTION RECALLED FROM HACS
 STATE FILE, VALUE IS 0 FOR ALL, OTHERWISE IS
 SYSTEM NUMBER (1=CGS, 2=SI, 3=ENG, 4=MXD)
 IPRAC = CONTROL OPTION TO SUPPRESS (0) OR TO SELECT (1)
 ACCESS TO THE HACS PHYSICAL PROPERTY DATA TAPE
 IPRRP = CONTROL OPTION TO SUPPRESS (0) OR TO SELECT (1)
 OUTPUT AUDIT OF HACS PHYSICAL PROPERTIES IF READ
 FROM PROPERTY TAPE
 NOFF = CONTROL OPTION TO SUPPRESS (0) OR TO SELECT (1)
 PRODUCTION OF DATA TAPE FOR OFF-LINE PLOTTING.
 IF SELECTED, DATA IS ACTUALLY GENERATED ONLY IF

```

C          CERTAIN RATE MODELS ARE EXECUTED AND THE USER HAS
C          REQUESTED PLOTTED OUTPUT OPTIONS FOR THESE MODELS.
C          NPRRP = INTEGER SWITCH USED IN DATA BASE SAVE AND RECALL
C          ROUTINES (FCNV ONLY) TO SELECT OPTIONAL OUTPUT
C          DURING PROPERTY FILE INTERFACE
C
C          COMMON/MODCN/MODEX(29),MODIO,LOCIO
C
C          COMMON/NAME/PTLST(30),SOURC(7)
C          INTEGER      PTLST
C
C          PTLST = DATA LIST OF ALL PATH CODES RECOGNIZED BY HACS, A TO
C                   Z, II, RR, SS AND BLANK, EACH STORED LEFT JUSTIFIED
C                   IN AN INTEGER WORD (A1 FORMAT). THE SEQUENCE IN
C                   WHICH THE PATH CODES ARE STORED IS USED TO VERIFY
C                   THE SEQUENCE OF PATH CODES SPECIFIED BY THE USER.
C                   E.G., MODEL A MUST BE EXECUTED BEFORE MODEL B IF
C                   BOTH ARE SPECIFIED.
C          SOURC = ARRAY OF LABELS DEFINING SOURCE CODES FOR FIELD VALUE
C                   DISPLAYS. LABELS ARE STORED IN AN ARRAY INDEXED
C                   FROM 1 TO 7, CORRESPONDING TO SOURCE CODES INDEXED
C                   FROM 0 TO 6
C
C          COMMON/OVCNT/MODNO,OVLST(29),SGLST(29)
C          INTEGER      OVLST,SGLST
C
C          MODNO = INTEGER INDEX FOR NEXT RATE MODEL TO BE EXECUTED,
C                   CORRESPONDS TO POSITION OF PATH CODE LETTER IN
C                   ARRAY PTLST AND HAS VALUES FROM 1 TO 30
C          OVLST = SPECIFIED OVERLAY NUMBER CONTAINING CODE FOR ALL
C                   VALID RATE MODELS AS OVLST(MODNO) WHERE MODNO IS
C                   IN RANGE 1 TO 29
C          SGLST = SPECIFIES SEGMENT NUMBER WITHIN OVERLAY FOR MODEL AS
C                   SGLST(MODNO) WHERE MODNO IS IN RANGE 1 TO 29. A
C                   VALUE OF 0 IS STORED FOR ALL UNDEFINED SEGMENTS.
C
C          COMMON/PLTCN/ANG,IBUF(4000),IFRST,IPLT,WIND
C
C          ANG    = SPECIFIES WIND DIRECTION FROM NORTH FOR USE IN OUTPUT
C                   LABEL ON OFF-LINE PLOT (NO LONGER IN USE). VALUE
C                   OF 0.0 IS USED INSTEAD TO SUPPRESS COMPASS LABEL.
C          IBUF   = ARRAY USED BY OFF-LINE PLOT ROUTINES AS A WORK AREA
C                   FOR BUILDING TAPE RECORDS
C          IFRST  = CONTROL SWITCH SET TO ONE TO EXECUTE PLOT
C                   INITIALIZATION ROUTINES ON FIRST PASS ONLY.
C          IPLT   = FILE NAME USED FOR PLOT TAPE
C          WIND   = WIND VELOCITY OBTAINED FROM HACS DATA FIELD 2016
C
C          EXTERNAL FCHCK
C          INTEGER FLDTAB(257)
C          INTEGER SCNTAB(32)
C          INTEGER MODTAB(30)
C          LOGICAL YESNO
C          INTEGER ERRAY(6)
C          DATA ERRAY/6*(-0)/,ERRAY(4)/0/
C          DATA FBLNK/10H
C
C          C-----INITIALIZE VARIABLES IN COMMON STORAGE FOR RE-EXECUTION
C          CALL OPENMS(11,FLDTAB,257,0)
C          CALL OPENMS(12,SCNTAB,32,0)
C          CALL OPENMS(13,MODTAB,30,0)
C          REWIND 10
C          CALL TRACE(0,0,0)
C          NOV = 0
C          SEG = 0

```

```

      CALL ERR(FCHCK)
      CALL SYSTEMC(115,ERRAY)
      ANG=0.0
      EOFF=0
      IFRST=1
      PLTYP=0
      STCON=1
      SVCON=1
C
C-----INITIALIZE OUTPUT PAGING ROUTINE
      CALL DATE(DTE)
      CALL TIME(TIM)
      WRITE(6,1020) DTE,TIM
10200FORMAT(/,5X,33HHAZARD ASSESSMENT COMPUTER SYSTEM/
1  5X,21HEXECUTION STARTED ON ,A10,4H AT ,A9//)
      LNCT=0
      NPG=0
      DO 5 I=1,10
5  TITLE(I)=FBLNK
      DO 6 I=1,29
6  MODEX(I)=0
      CALL PAGER(5)
      LOCIO=2
C
C-----RETURN HERE TO ACCESS HACS DATA INPUT PROCESSOR. INITIAL CALL
C      TO FAULT CHECK ROUTINE INITIALIZES ERROR INDICATORS BEFORE
C      ENTERING OVERLAY. INPUT PROCESSOR RETURNS NOP = 0 TO TERMINATE
C      OR NOP = 1,2 OR 3 TO EXECUTE ASSESSMENT RUN.
10 CALL OVLOD(1)
C
C-----SKIP TO END IF USER OPTION CANCELLED, OR DOES NOT REQUIRE
C      EXECUTION OF RATE MODELS
      IF(NOP.EQ.0) GO TO 40
C
C-----LOOP ON RATE MODEL INDICES OBTAINED FROM USER PATH CODE
C      INPUT UNTIL INDEX CORRESPONDING TO FIRST BLANK IS FOUND.
C      EXECUTE OVERLAY FOR EACH MODEL.
      DO 20 I=1,15
      MODNO=MODEL(I)
      IF(MODNO.GE.30) GO TO 30
      NOV=OVLST(MODNO)
      IF(NOV.EQ.0) GO TO 20
15  MODIO=MODEX(MODNO)
      CALL OVLOD(NOV)
      MODEX(MODNO)=1
      STCON=4
C
C-----TEST FOR SELECTION OF OFF-LINE PLOTTED OUTPUT
C      IF(NOFF.EQ.0) GO TO 20
C      IF(PLTYP.EQ.0) GO TO 20
C
C-----EXECUTE OFF-LINE PLOTTED OUTPUT POST-PROCESSOR
C      NOV = 2
C      CALL OVLOD(2)
C      CALL PAGER(3)
C      WRITE(LP,1010)
C      WRITE(LP,1030)
      IF(YESNO(0)) GO TO 15
1030 FORMAT (34H DO YOU WANT TO RE-RUN THIS MODEL?)
20 CONTINUE
C
C-----TEST FOR FILE DISPLAY OPTION AFTER MODEL EXECUTION
30 CALL LSTFL(3)
      CALL SUMRY
C
C-----RETURN TO INPUT DATA PROCESSOR IF AN END OF FILE HAS NOT
C      YET BEEN ENCOUNTERED, OTHERWISE TERMINATE RUN.
      IF(EOFF.EQ.0) GO TO 10
C
C-----END OF RUN
40 CALL PAGER(5)
      WRITE(LP,1000)

```



```

INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
DATA BLANK/1H /,IN/5/,NO/1HN/,OUT/6/,YES/1HY/
ODATA (SPLST(I),I=1,14)/1H0,1H1,1H2,1H3,1H4,
1 1H5,1H6,1H7,1H8,1H9,1H+,1H-,1H.,1HE/

```

```

C
C-----IDENTIFY PLOT OUTPUT FILE
C

```

```
DATA IPLT/4LIPLT/
```

```

C
C-----INITIALIZE SAVE FILE
C

```

```

ODATA (MSG(I),I=1,10)/10*10H /,MNF/225/,MNI/50/,NF/0/,
1 NIS/1/,LISTS(1,1)/1001/,LISTS(1,2)/460/,(LISTS(1,1),I=3,5)
2 /3*4H /,LISTS(1,6)/1/,IVAL(1,1)/4H /,IVAL(1,2)
3 /4HN/A /,IVAL(1,3)/4HN/A /

```

```

C
C-----INITIALIZE STATE FILE
C

```

```

ODATA (MSG(I),I=1,10)/10*10H /,MNF/225/,MNI/50/,NF/0/,
1 NI/1/,LIST(1,1)/1001/,LIST(1,2)/460/,(LIST(1,1),I=3,5)/3*
2 4H /,LIST(1,6)/1/,IVAL(1,1)/4H /,IVAL(1,2)/4HN/A /,
3 IVAL(1,3)/4HN/A /

```

```

C
C-----INITIALIZE PATH CODE INPUT ARRAY
C

```

```
DATA (UPTH(I),I=1,15)/15*4H /
```

```

C
C-----INITIALIZE AND SET DATA FOR OPERATIONS CONTROL
C

```

```

ODATA EOFF/0/,IDFLT/10/,LBL(1)/8H EMPTY /,LBL(2)/8HDEFAULT /,
1 LBL(3)/8H USER /,LBL(4)/8HCOMPUTED/,STCON/1/,SVCON/1/

```

```

C
C-----SET DATA FOR UNIT CONVERSIONS. NOTE THAT ADDITIONAL DATA IS
C DEFINED AS NECESSARY IN SUBROUTINES FOR USE IN CONVERTING
C TEMPERATURE VALUES (TYPE 06). ALL OTHER CONVERSIONS ARE
C APPLIED AS SCALE FACTORS.

```

```
DATA MSYS/4/,MTYP/47/
```

```

C
ODATA
1 (CONV(I, 1),I=1,3)/1.0 ,1.0 ,1.0 /,
2 (CONV(I, 2),I=1,3)/100. ,30.48 ,160900.0 /,
3 (CONV(I, 3),I=1,3)/1000000.0 ,28317.0 ,3786.09 /,
4 (CONV(I, 4),I=1,3)/0.001 ,0.01602 ,1.0 /,
5 (CONV(I, 5),I=1,3)/10.0 ,68950.0 ,1333.0 /,
6 (CONV(I, 6),I=1,3)/273.15 ,32.0 ,273.15 /,
7 (CONV(I, 7),I=1,3)/10000.0 ,929.0304 ,10000.0 /,
8 (CONV(I, 8),I=1,3)/1000.0 ,453.6 ,907200.0 /,
9 (CONV(I, 9),I=1,3)/0.0002389 ,1.0 ,1.0 /

```

```

C
ODATA
1 (CONV(I, 10),I=1,3)/0.0002389 ,0.55556 ,0.0002389 /,
2 (CONV(I, 11),I=1,3)/1.0 ,60.0 ,3600.0 /,
3 (CONV(I, 12),I=1,3)/1000.0 ,453.6 ,252.0 /,
4 (CONV(I, 13),I=1,3)/1.0 ,0.01745 ,0.01745 /,
5 (CONV(I, 14),I=1,3)/0.00002389 ,0.0000753 ,0.00002778 /,
6 (CONV(I, 15),I=1,3)/100.0 ,44.7 ,51.44 /,
7 (CONV(I, 16),I=1,3)/1.0 ,1.0 ,1.0 /,
8 (CONV(I, 17),I=1,3)/0.002389 ,0.004134 ,0.002778 /,
9 (CONV(I, 18),I=1,3)/1000.0 ,453.6 ,1000.0 /

```

```

C
ODATA
1 (CONV(I, 19),I=1,3)/10000.0 ,929.0304 ,10000.0 /,
2 (CONV(I, 20),I=1,3)/1.0 ,0.01667 ,0.0002778 /,
3 (CONV(I, 21),I=1,3)/1.0 ,1.0 ,1.0 /,
4 (CONV(I, 22),I=1,3)/10. ,0.01 ,0.01 /,
5 (CONV(I, 23),I=1,3)/1.0 ,1.0 ,1.0 /,
6 (CONV(I, 24),I=1,3)/1.0 ,1.0 ,1.0 /,
7 (CONV(I, 25),I=1,3)/0.1 ,0.4883 ,0.1 /,
8 (CONV(I, 26),I=1,3)/1.0 ,1.0 ,1.0 /

```

C	9 (CONV(I, 27), I=1,3)/1.0	,1.0	,0.001	/	
	ODATA				
	1 (CONV(I, 28), I=1,3)/0.0002389	,1.0	,1.0	/,	
	2 (CONV(I, 29), I=1,3)/1.0	,1.0	,1.0	/,	
	3 (CONV(I, 30), I=1,3)/1.0	,1.0	,1.0	/,	
	4 (CONV(I, 31), I=1,3)/1.0	,1.0	,1.0	/,	
	5 (CONV(I, 32), I=1,3)/1.0	,1.0	,1.0	/,	
	6 (CONV(I, 33), I=1,3)/1.0	,1.0	,1.0	/,	
	7 (CONV(I, 34), I=1,3)/1.0	,1.0	,1.0	/,	
	8 (CONV(I, 35), I=1,3)/1.0	,1.0	,1.0	/,	
	9 (CONV(I, 36), I=1,3)/1.0	,1.0	,1.0	/	
C	ODATA				
	1 (CONV(I, 37), I=1,3)/1.0	,1.0	,1.0	/,	
	2 (CONV(I, 38), I=1,3)/1.0	,1.0	,1.0	/,	
	3 (CONV(I, 39), I=1,3)/1.0	,1.0	,1.0	/,	
	4 (CONV(I, 40), I=1,3)/1.0	,1.0	,1.0	/,	
	5 (CONV(I, 41), I=1,3)/1.0	,1.0	,1.0	/,	
	6 (CONV(I, 42), I=1,3)/1.0	,1.0	,1.0	/,	
	7 (CONV(I, 43), I=1,3)/1.0	,1.0	,1.0	/,	
	8 (CONV(I, 44), I=1,3)/0.0075	,51.725	,0.00075	/,	
	9 (CONV(I, 45), I=1,3)/10.0	,68950.0	,10000000.0	/	
C	ODATA				
	1 (CONV(I, 46), I=1,3)/1.0	,1.0	,1.0	/,	
C	2 (CONV(I, 47), I=1,3)/100.0	,0.042333	,0.001667	/	
	ODATA				
	1 (UNIT(I, 1), I=1,4)/8HND	,8HND	,8HND	,8HND	/,
	2 (UNIT(I, 2), I=1,4)/8HCM	,8HM	,8HFT	,8HMI	/,
	3 (UNIT(I, 3), I=1,4)/8HCM3	,8HM3	,8HFT3	,8HGALS	/,
	4 (UNIT(I, 4), I=1,4)/8HG/CM3	,8HKG/M3	,8HLB/FT3	,8HG/CM3	/,
	5 (UNIT(I, 5), I=1,4)/8HD/CM2	,8HN/M2	,8HPSI	,8HMM HG	/,
	6 (UNIT(I, 6), I=1,4)/8HC	,8HK	,8HF	,8HK	/,
	7 (UNIT(I, 7), I=1,4)/8HCM2	,8HM2	,8HFT2	,8HM2	/,
	8 (UNIT(I, 8), I=1,4)/8HG	,8HKG	,8HLB	,8HTN	/,
	9 (UNIT(I, 9), I=1,4)/8HCL/GC	,8HJ/KGK	,8HBT/LBF	,8HCL/GK	/
C	ODATA				
	1 (UNIT(I, 10), I=1,4)/8HCL/G	,8HJ/KG	,8HBT/LB	,8HJ/KG	/,
	2 (UNIT(I, 11), I=1,4)/8HS	,8HS	,8HMIN	,8HHR	/,
	3 (UNIT(I, 12), I=1,4)/8HG/S	,8HKG/S	,8HLB/S	,8HTN/HR	/,
	4 (UNIT(I, 13), I=1,4)/8HRAD	,8HRAD	,8HDEG	,8HDEG	/,
	5 (UNIT(I, 14), I=1,4)/8HCL/CM2S	,8HM/M2	,8HBT/FT2H	,8HCC/M2H	/,
	6 (UNIT(I, 15), I=1,4)/8HCM/S	,8HM/S	,8HMPH	,8HKNOTS	/,
	7 (UNIT(I, 16), I=1,4)/8HPPM	,8HPPM	,8HPPM	,8HPPM	/,
	8 (UNIT(I, 17), I=1,4)/8HCL/CMSC	,8HM/MK	,8HBT/FTHF	,8HCC/MHK	/,
	9 (UNIT(I, 18), I=1,4)/8HD/CM	,8HN/M	,8HLB/S2	,8HN/M	/
C	ODATA				
	1 (UNIT(I, 19), I=1,4)/8HCM2/S	,8HM2/S	,8HFT2/S	,8HM2/S	/,
	2 (UNIT(I, 20), I=1,4)/8H/S	,8H/S	,8H/MIN	,8H/HR	/,
	3 (UNIT(I, 21), I=1,4)/8HG/HG	,8HKG/HKG	,8HLB/HLB	,8HG/HG	/,
	4 (UNIT(I, 22), I=1,4)/8HDS/CM2	,8HNS/M2	,8HCP	,8HCP	/,
	5 (UNIT(I, 23), I=1,4)/8HG/GM	,8HKG/KGM	,8HLB/LBM	,8HKG/KGM	/,
	6 (UNIT(I, 24), I=1,4)/8HLOG FCN	,8HLOG FCN	,8HLOG FCN	,8HLOG FCN	/,
	7 (UNIT(I, 25), I=1,4)/8HG/CM2S	,8HKG/M2S	,8HLB/FT2S	,8HKG/M2S	/,
	8 (UNIT(I, 26), I=1,4)/8HPERCENT	,8HPERCENT	,8HPERCENT	,8HPERCENT	/,
	9 (UNIT(I, 27), I=1,4)/8HG/G	,8HKG/KG	,8HLB/LB	,8HG/KG	/
C	ODATA				
	1 (UNIT(I, 28), I=1,4)/8HCL/GMC	,8HJ/KGMK	,8HBT/LBMF	,8HCL/GMK	/,
	2 (UNIT(I, 29), I=1,4)/8HG/CM3	,8HG/CM3	,8HG/CM3	,8HG/CM3	/,
	3 (UNIT(I, 30), I=1,4)/8HG/CM3C	,8HG/CM3C	,8HG/CM3C	,8HG/CM3C	/,
	4 (UNIT(I, 31), I=1,4)/8HG/CM3C2	,8HG/CM3C2	,8HG/CM3C2	,8HG/CM3C2	/,
	5 (UNIT(I, 32), I=1,4)/8HLN FCN	,8HLN FCN	,8HLN FCN	,8HLN FCN	/,
	6 (UNIT(I, 33), I=1,4)/8HC	,8HC	,8HC	,8HC	/,
	7 (UNIT(I, 34), I=1,4)/8HCL/CMSC	,8HCL/CMSC	,8HCL/CMSC	,8HCL/CMSC	/,
	8 (UNIT(I, 35), I=1,4)/8HCL/CMSC2	,8HCL/CMSC2	,8HCL/CMSC2	,8HCL/CMSC2	/,
	9 (UNIT(I, 36), I=1,4)/8HCL/GC	,8HCL/GC	,8HCL/GC	,8HCL/GC	/
C					

C
C
C
C

THIS SUBROUTINE INDICATES THE COMPLETION OF A MODELS EXECUTION
SUBROUTINES REQUIRED - PAGER

CALL PAGER(3)
WRITE(6,100) NAME
RETURN
100 FORMAT(//23H THE EXECUTION OF MODEL,A4,14H IS COMPLETED.)
END
SUBROUTINE EPRNT(MOD,IS,IR,IFLAG)

C
C
C
C
C
C
C
C
C
C
C
C

SUBROUTINE EPRNT PRODUCES ERROR MESSAGES FOR A GIVEN MODEL

MOD = NAME OF MODEL
IS = LOWEST SOURCE CODE OBTAINED IN A SERIES OF RETRIEVALS
FROM THE HACS STATE FILE
IR = ERROR STATUS OF SERIES OF STATE FILE RETRIEVALS, SET
TO 1 ON INPUT IF AT LEAST ONE ERROR OCCURRED
IFLAG = OUTPUT INDICATOR, SET TO 0 IF MODEL IS NOT IN ERROR
MODE, SET TO 1 IF ERROR RETURN IS TO BE EXECUTED

SUBROUTINES REQUIRED - PAGER

COMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
INTEGER UPTH
REAL MSG
DIMENSION STATE(2489)
EQUIVALENCE (STATE(1),MSG(1))
COMMON/MODCN/MODEX(29),MODID,LOCIO
LOGICAL ENTR, INTEGR,YESNO
IF(LOCIO.NE.1) GO TO 2
1 LOCIO=20
IFLAG=2
RETURN
2 CONTINUE
IFLAG=0
IF(IS.NE.0) GO TO 10
CALL PAGER(1)
WRITE(6,100) MOD
GO TO 30
10 IF(IS.NE.1) GO TO 20
CALL PAGER(1)
WRITE(6,102) MOD
20 IF(IR.NE.1) GO TO 40
CALL PAGER(1)
WRITE(6,101) MOD
30 IFLAG=1
CALL PAGER(1)
WRITE(6,103)
RETURN
40 WRITE(6,104)
104 FORMAT (40H DO YOU WISH TO CHANGE ANY MODEL INPUTS?)
IF(YESNO(0)) GO TO 50
LOCIO=2
IFLAG=0
RETURN
50 WRITE(6,105)
105 FORMAT (20H ENTER FIELD NUMBER:)
GO TO 52
51 WRITE(6,107)
107 FORMAT (35H ENTER FIELD NUMBER OR 9999 TO EXIT)
52 IF(.NOT.ENTR(0)) GO TO 51
IF(.NOT.INTEGR(IFLD)) GO TO 51
IF(IFLD.EQ.9999) GO TO 65
NFLD=NI+NF
DO 53 I=1,NFLD
IFLN=I
IF(IFLD.EQ.LIST(I,1)) GO TO 60
55 CONTINUE
WRITE(6,106) IFLD

```

GO TO 51
60 IVAR=LIST(ILN,2)/1000
   IERR=0
   ISRC=6
   LOCIO=3
   IF(IVAR.EQ.0) CALL IRCL(IFLD,IVL,ISRC,IERR)
   IF(IVAR.EQ.1) CALL FRCL(IFLD,VAL,ISRC,IERR)
GO TO 51
65 LOCIO=20
   IFLAG=2
   RETURN

```

ERR	IDENT	ERR
	ENTRY	ERR
	BSSZ	1
	SB1	1
	SB2	X1+1
	EREXIT	B2
	EQ	ERR
	END	

**AUTHOR - R.S. RIDGLEY, CONTROL DATA CORP.
6003 EXECUTIVE BLVD.
ROCKVILLE, MD. 20852
(301)468-8166**

```
COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)
COMMON/OVER/NOV, SEG
DIMENSION ERROR(12,2), ATHARR(8,2)
```

```

C  FETCH R40 CONTAINING ERROR INFORMATION
C  IRA=MEMGET(0)
C  MASK OFF ERROR FLAG IN BITS 24-29 AND RIGHT JUSTIFY
C  IEF=AND(IRA,77000000000B)
C  IEFL=SHIFT(IEF,-24)

```

```

C   MASK OFF ERROR MODE IN BITS 48-53 AND RIGHT JUSTIFY
    IEM=AND(IRA,770000000000000000B)
    IEMD=SHIFT(IEM,-48)
C   EXTRACT ADDRESS OF ERROR
    IADD = AND(IRA,77777700000000000B)
    IADD = SHIFT(IADD,-30)
    CALL PAGER(1)
    WRITE(LP,1000)NDV,SEG,(ERRAR(IEFL,I),I=1,2)
    IF(IEFL.EQ. 2)WRITE(LP,1001)(ATHARR(IEMD+1,I),I=1,2)
    WRITE(LP,1002)IADD,IADD
    ENDFILE 61
    ENDFILE 62
1000  FORMAT(12H IN OVERLAY ,I2,9H,SEGMENT ,I2,5H,*** ,2A10)
1001  FORMAT(31X,4H*** ,2A10)
1002  FORMAT(/* ADDRESS OF ERROR IS :(DECIMAL)- *,I7,* (OCTAL) *
      +,O10)
      STOP
      END
      SUBROUTINE FCNV(IFLD,ILN,VAL,ITYP,IS)

```

```

      SUBROUTINE FCNV IS USED TO GENERATE DISPLAYS OF REAL FIELD
      VALUES IN ALTERNATE SYSTEMS OF UNITS OF MEASURE AS PART OF
      THE HACS DATA BASE SAVE AND RECALL FUNCTIONS.  OUTPUT PRODUCED
      BY THIS ROUTINE IS CONTROLLED BY THE OPTION ICVSL TO DISPLAY
      FIELD VALUES IN A SINGLE SPECIFIED SYSTEM OF UNITS, OR IN
      ALL UNIQUE UNITS DEFINED FOR THE FIELD.

```

```

      FAC      = ARRAY OF UNIT CONVERSION FACTORS FOR TEMPERATURES
      I        = INDEX ON UNIT SYSTEMS FROM 1 TO MSYS
      IFLD     = ARGUMENT, FIELD NUMBER
      ILN      = ARGUMENT, INDEX TO FIELD NAME IN STATE FILE
      IS       = ARGUMENT, INDEX (-1) TO SOURCE CODE LABEL
      ITYP     = ARGUMENT, QUANTITY TYPE CORRESPONDING TO FIELD VALUE
                TO BE CONVERTED FOR OUTPUT DISPLAY
      J        = INDEX RANGING FROM 1 TO I-1 USED TO DETECT DUPLICATE
                UNIT LABELS, ALSO INDEX FOR NAME OUTPUT
      JJ       = INTEGER SELECTOR SET TO SYSTEM OF UNITS FOR WHICH
                A FULL AUDIT LINE IS PRINTED
      TAG      = TEMPORARY STORAGE FOR UNIT LABEL IN SYSTEM I, FOR
                QUANTITY TYPE ITYP
      VAL      = ARGUMENT, FIELD VALUE IN INTERNAL SYSTEM OF UNITS
      XVAL     = FIELD VALUE CONVERTED TO UNIT SYSTEM I FOR OUTPUT

```

```

      COMMON VARIABLES USED - CONV,ICVSL,LIST,LP,MSYS,NPRRP,SOURC,
                           UNIT

```

```

      SUBROUTINES REQUIRED - PAGER

```

```

      AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
                35/309A ACORN PARK,
                CAMBRIDGE, MASS., 02140
                TEL. 617-864-5770 EXT. 2813

```

```

      DATE - 29 JANUARY 1976

```

```

0COMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1  NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
  INTEGER      UPTH
  REAL         MSG
  DIMENSION    STATE(2489)
  EQUIVALENCE (STATE(1),MSG(1))

```

```

COMMON/CNVDT/CONV(3,47),MSYS,HTYP,UNIT(4,47)

```

```

COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)

```

```

COMMON/IDCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP

```

```

COMMON/NAME/PTLST(30),SOURC(7)
  INTEGER      PTLST

```

```

C      DIMENSION FAC(3)
C      DATA      (FAC(I),I=1,3)/1.0,1.8,1.0/,FBLNK/8H
C
C-----SUPPRESS DISPLAY IF NPRRP SWITCH IS SET TO 0 FOR SUPPRESSING
C      AUDIT OF PROPERTY FILE ACCESS
C      IF(NPRRP.EQ.0) RETURN
C
C-----INITIALIZE LOOP USING INDEX I ON SYSTEM OF UNITS SELECTED
C      OR TO FIRST SYSTEM IF ICVSL SET TO 0 FOR ALL SYSTEMS.
C      I=ICVSL
C      IF(I.EQ.0) I=1
C
C-----SET INDICATOR JJ TO CONTROL OUTPUT FORMAT FOR FIRST LINE AND
C      RETRIEVE LABEL TAG FOR FIRST LINE VALUE
C      JJ=I
C      TAG=UNIT(I,ITYP)
C
C-----CONVERT FIELD VALUE IN SYSTEM 1 (INTERNAL) TO SELECTED
C      SYSTEM FOR OUTPUT DISPLAY. BRANCH ON TYPE OF PHYSICAL
C      QUANTITY TO SELECT CONVERSION EQUATION.
C      IF(I.NE.1) GO TO 20
C      XVAL=VAL
C      GO TO 40
C      20 IF(ITYP.EQ.6) GO TO 30
C      XVAL=VAL/CONV(I-1,ITYP)
C      GO TO 40
C      30 XVAL=CONV(I-1,ITYP)+VAL*FAC(I-1)
C
C-----DISPLAY CONVERTED VALUE AS PART OF AUDIT OUTPUT
C      40 CALL PAGER(1)
C      01 IF(I.EQ.JJ) WRITE(LP,1020) IFLD,(LIST(ILN,J),J=3,5),XVAL,TAG,
C      1      SOURC(IS+1)
C      IF(I.NE.JJ) WRITE(LP,1000) XVAL,TAG
C
C-----TEST FOR COMPLETION ON SINGLE UNIT OPTION OR ALL UNIQUE
C      UNITS DISPLAYED.
C      IF(ICVSL.NE.0) GO TO 70
C      50 I=I+1
C      IF(I.GT.MSYS) GO TO 70
C
C-----AUDIT VALUE WHEN OPTION FOR ALL SYSTEMS SELECTED ONLY USING
C      EACH UNIQUE UNIT. OMIT DUPLICATES. NOTE THAT I CANNOT BE 1.
C      TAG=UNIT(I,ITYP)
C      J=1
C      60 IF(TAG.EQ.UNIT(J,ITYP)) GO TO 50
C      J=J+1
C      IF(J.LT.I) GO TO 60
C      GO TO 20
C
C-----INSERT SPACE BETWEEN DISPLAYS FOR DIFFERENT FIELD VALUES ON
C      OUTPUT AUDIT.
C      70 CALL PAGER(1)
C      WRITE(LP,1010)
C      RETURN
C
C      1000 FORMAT (24X,3H= ,G13.4,2X,A8)
C      1010 FORMAT (5X)
C      1020 FORMAT (5X,I4,1X,3A4,5H = ,G13.4,2X,A8,8H, IS A ,A8,7H VALUE)
C      END
C      SUBROUTINE FRCL(IFLD,VAL,ISRC,IERR)
C
C      SUBROUTINE FRCL RECALLS THE VALUE (VAL) OF A REAL FIELD,
C      DEFINED BY THE FIELD NUMBER IFLD, FROM THE HACS STATE FILE.
C      ERROR CONDITIONS WILL PRODUCE MESSAGES, AND CAUSE VALUES OF
C      VAL=0.0, IERR=1 AND ISRC=0 TO BE RETURNED. ARGUMENTS ISRC AND
C      IERR ARE INITIALIZED AND TESTED IN A CALLING PROGRAM TO
C      DETERMINE THE STATUS OF A GROUP OF DATA BASE RECALL OPERATIONS.
C
C      I      = INTEGER FORTRAN ARRAY INDEX

```

```

C      IERR = ERROR CONDITION INDICATOR SET TO 1 IF ANY ERROR IS
C      DETECTED IN PERFORMING THE REQUESTED RECALL
C      IFLD = ARGUMENT, FIELD NUMBER FOR WHICH VALUE IS REQUESTED
C      ILN  = INDEX INTO STATE FILE TO OBTAIN DEFINITION OF FIELD
C      IS   = SOURCE CODE FOR VALUE CURRENTLY STORED IN STATE FILE
C      ISRC = ARGUMENT, MINIMUM SOURCE CODE FOUND IN ONE OR MORE
C      RECALL OPERATIONS GROUPED FOR INPUT TO A RATE MODEL
C      ITYP = PRE-DEFINED TYPE OF PHYSICAL QUANTITY FOR FIELD IFLD
C      IVAR = TYPE OF INTERNAL FIELD STORAGE FOR FIELD IFLD
C      (0 FOR INTEGER, 1 FOR REAL)
C      IX   = INDEX STORED IN STATE FILE TO LINK FIELD DEFINITION
C      TO FIELD VALUE ARRAYS
C      NFLD = (NI+NF) GIVES THE TOTAL NUMBER OF FIELD DEFINITIONS
C      ACTUALLY STORED IN THE STATE FILE
C      VAL  = ARGUMENT, RETURNED AS VALUE OF FIELD IFLD, AS RECALLED
C      FROM THE STATE FILE IN INTERNAL UNITS, OR 0.0 IF
C      AN ERROR WAS ENCOUNTERED.

COMMON VARIABLES USED - FVAL,LIST,LP,NF,NI
SUBROUTINES REQUIRED - FCNV,PAGER
AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
        35/309A ACORN PARK,
        CAMBRIDGE, MASS., 02140
        TEL. 617-864-5770 EXT. 2813
DATE - 29 JANUARY 1976

COMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
  INTEGER UPTH
  REAL MSG
  DIMENSION STATE(2489)
  EQUIVALENCE (STATE(1),MSG(1))

COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)

COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
COMMON/MODCN/MODEX(29),MODIO,LOCIO

COMMON/NAME/PTLST(30),SOURC(7)
  INTEGER PTLST

COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP
LOGICAL QUEST
LOGICAL YESNO,ENTR,FLTPT,NAME
DIMENSION FAC(3)
DATA (FAC(I),I=1,3)/1.0,1.8,1.0/

C-----LOOP THROUGH ALL FIELD DEFINITIONS IN STATE FILE TO LOCATE
C      FIELD NUMBER IFLD. NOTE THAT AT LEAST ONE FIELD (FOR CHEMICAL
C      RECOGNITION CODE) BY DEFINITION HAS BEEN PREVIOUSLY DEFINED.
C      THE VARIABLE ILN SAVES THE LOCATION IN THE STATE FILE
C      CORRESPONDING TO FIELD NUMBER IFLD, IF FOUND.
C      NFLD=NI+NF
C      DO 10 I=1,NFLD
C      ILN=1
C      IF(IFLD.EQ.LIST(I,1)) GO TO 20
C      10 CONTINUE

C-----ERROR. THE REQUESTED FIELD NUMBER DOES NOT EXIST IN THE HACS
C      STATE FILE. THIS CONDITION INDICATES EITHER AN ERROR IN
C      PROGRAM CODING OF CALLS TO SUBROUTINE FRCL, OR A MISSING
C      DEFAULT FILE SPECIFICATION TO DEFINE THE FIELD BEING REQUESTED.
C      CALL PAGER(2)
C      WRITE(LP,1000) IFLD
C      GO TO 60

C-----VERIFY REQUEST FOR REAL VARIABLE WITH STORAGE MODE OF VARIABLE
C      IN STATE FILE.

```

```

20 IVAR=LIST(ILN,2)/1000
   IF(IVAR.EQ.1) GO TO 30
C-----
C-----ERROR. FLOATING POINT VALUE HAS BEEN REQUESTED FOR VARIABLE
C-----DEFINED IN STATE FILE AS AN INTEGER.
      CALL PAGER(2)
      WRITE(LP,1010) IFLD,(LIST(ILN,1),I=3,5)
      GO TO 60
C-----
C-----FIELD TYPE IS REAL, UNPACK QUANTITY TYPE AND SOURCE CODE OF
C-----STORED VALUE.
      30 IS=1000*IVAR
      ITYP=(LIST(ILN,2)-IS)/10
      IS=LIST(ILN,2)-10*ITYP-IS
C-----
C-----UPDATE SUBROUTINE ARGUMENT TO TRACK LOWEST SOURCE CODE OF
C-----FIELD VALUES RECALLED FOR USE IN EXECUTING A RATE MODEL.
      IF(IS.LT.ISRC) ISRC=IS
C-----
C-----IF FIELD VALUE HAS NOT BEEN DEFINED, SET ERROR FLAG AND USE
C-----0.0 FOR STANDARD AUDIT.
      IF(IS.GT.0) GO TO 40
      IERR=1
      VAL=0.0
      GO TO 50
C-----
C-----INDEX INTO DATA ARRAY TO RETURN VALUE OF REQUESTED FIELD
      40 IX=LIST(ILN,6)
      VAL=FVAL(IX,1)
C-----
C-----GENERATE HACS AUDIT DISPLAY FOR FIELD DEFINITION AND CURRENT
C-----VALUES IN ALL DEFINED UNIT SYSTEMS.
      50 IF(LOCIO.NE.2) GO TO 100
C-----SECTION FOR LOCIO = 2, RECALL AND
C-----DISPLAY FOR MODEL SUMMARY.
      51 CALL FCNV(IFLD,ILN,VAL,ITYP,IS)
      IF(LOCIO.NE.2) GO TO 190
C-----MIN/MAX TEST
      99 FMN=FVAL(IX,2)
      FMX=FVAL(IX,3)
      IF(VAL.LT.FMN) GO TO 260
      IF(VAL.LE.FMX) GO TO 270
      260 WRITE(LP,2040) VAL,FMN,FMX,UNIT(1,ITYP)
      2040FORMAT (25H WARNING, FIELD VALUE OF ,G13.4,11H NOT WITHIN /
      1 18H NOMINAL RANGE OF ,G13.4,4H TO ,G13.4,4H IN ,A8)
      270 CONTINUE
      GO TO 280
      100 IF(LOCIO.EQ.3) GO TO 105
      IF(IS.GT.1) GO TO 280
      105WRITE(LP,110) (LIST(ILN,I),I=3,5),UNIT(1,ITYP),UNIT(2,ITYP),
      1 UNIT(3,ITYP),UNIT(4,ITYP)
      110FORMAT (22H ENTER REAL VALUE FOR ,3A4,
      1 4H IN ,A8,1H,A8,1H,A8,1H,A8)
      IF(.NOT.ENTR(0)) GO TO 51
      IF(.NOT.QUEST(0)) GO TO 199
      CALL EXPLAIN(ILN)
      GO TO 105
      199 CONTINUE
      IF(.NOT.FLTPT(FVL)) GO TO 105
      ISYS=1
      IF(.NOT.NAME(TAG)) GO TO 220
      200 IF(TAG.EQ.UNIT(ISYS,ITYP)) GO TO 220
      ISYS=ISYS+1
      IF(ISYS.LE.4) GO TO 200
      WRITE(LP,2020) TAG
      2020 FORMAT (13H UNIT LABEL / ,A10,12H/ IS INVALID)
      GO TO 105
      220 IF(ISYS.EQ.1) GO TO 250
      JSYS=ISYS-1
      D=CONV(JSYS,ITYP)
      IF(ITYP.NE.6) GO TO 230
      FVL=(FVL-D)/FAC(JSYS)

```

```

      GO TO 240
230 FVL=D*FVL
240 CONTINUE
250 CONTINUE
C-----REPLACE STATE FILE VALUE
      FVAL(IX,1)=FVL
      VAL=FVL
      LIST(ILN,2)=LIST(ILN,2)-IS+5
      IS=5
      GO TO 99
190 WRITE(LP,195)
195 FORMAT (31H DO YOU WISH TO USE THIS VALUE?)
      IF(YESNO(0)) GO TO 99
      GO TO 105
280 IF(IS.LT.ISRC) ISRC=IS
      RETURN
60 IERR=1
      ISRC=0
      VAL=0.0
      RETURN
C
10000FORMAT (5X,26H*****ERROR - FIELD NUMBER ,I4,42H REQUESTED FOR RECA
11L HAS NOT BEEN DEFINED/)
10100FORMAT (5X,53H*****ERROR - REAL RECALL REQUESTED FOR INTEGER FIELD
1 ,I4,1X,3A4/)
      END
      SUBROUTINE FSV(IFLD,VAL,ISRC)

      SUBROUTINE FSV SAVES THE VALUE (VAL) OF A REAL FIELD, DEFINED
      BY THE FIELD NUMBER IFLD, IN THE HACS STATE FILE DEPENDING ON
      THE SOURCE CODE. THE NEW VALUE IS SAVED IF ITS SOURCE CODE
      (ISRC) IS GREATER THAN THE SOURCE CODE OF THE VALUE ALREADY
      STORED IN THE STATE FILE. WHETHER OR NOT THE VALUE IS SAVED,
      THE ROUTINE PRODUCES AN OUTPUT AUDIT.

      I      = INTEGER FORTRAN ARRAY INDEX
      IFLD   = ARGUMENT, FIELD NUMBER FOR WHICH VALUE IS TO BE SAVED
      ILN    = INDEX INTO STATE FILE TO OBTAIN DEFINITION OF FIELD
      IS     = SOURCE CODE FOR VALUE CURRENTLY STORED IN STATE FILE
      ISRC   = ARGUMENT, SOURCE CODE ASSOCIATED WITH VALUE OF FIELD
              TO BE SAVED
      ITYP   = PRE-DEFINED TYPE OF PHYSICAL QUANTITY FOR FIELD IFLD
      IVAR   = TYPE OF INTERNAL FIELD STORAGE FOR FIELD IFLD
              (0 FOR INTEGER, 1 FOR REAL)
      IX     = INDEX STORED IN STATE FILE TO LINK FIELD DEFINITION
              TO FIELD VALUE ARRAYS
      NFLD   = (NI+NF) GIVES THE TOTAL NUMBER OF FIELD DEFINITIONS
              ACTUALLY STORED IN THE STATE FILE
      VAL    = ARGUMENT, GIVES VALUE OF FIELD IFLD TO BE SAVED

      COMMON VARIABLES USED - FVAL,LIST,LP,NF,NI,NPRRP,SOURC,UNIT
      SUBROUTINES REQUIRED - FCNV,PAGER

      AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
              35/309A ACORN PARK,
              CAMBRIDGE, MASS., 02140
              TEL. 617-864-5770 EXT. 2813
      DATE   - 3 FEBRUARY 1976

      OCOMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1      NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
      INTEGER      UPTH
      REAL         MSG
      DIMENSION    STATE(2489)
      EQUIVALENCE (STATE(1),MSG(1))
C
      COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)
C
      COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)

```

```

C
COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP
C
COMMON/NAME/PTLST(30),SOURC(7)
INTEGER PTLST
C
C-----LOOP THROUGH ALL FIELD DEFINITIONS IN STATE FILE TO LOCATE
C FIELD NUMBER IFLD. NOTE THAT AT LEAST ONE FIELD (FOR CHEMICAL
C RECOGNITION CODE) BY DEFINITION HAS BEEN PREVIOUSLY DEFINED.
C THE VARIABLE ILN SAVES THE LOCATION IN THE STATE FILE
C CORRESPONDING TO FIELD NUMBER IFLD, IF FOUND.
NFLD=NI+NF
DO 10 I=1,NFLD
  ILN=I
  IF(IFLD,EQ,LIST(I,1)) GO TO 20
10 CONTINUE
C
C-----ERROR. THE REQUESTED FIELD NUMBER DOES NOT EXIST IN THE HACS
C STATE FILE. THIS CONDITION INDICATES EITHER AN ERROR IN
C PROGRAM CODING OF CALLS TO SUBROUTINE FRCL, OR A MISSING
C DEFAULT FILE SPECIFICATION TO DEFINE THE FIELD BEING REQUESTED.
CALL PAGER(2)
WRITE(LP,1000) IFLD
RETURN
C
C-----VERIFY REQUEST TO SAVE REAL VARIABLE WITH STORAGE MODE OF
C VARIABLE IN STATE FILE.
20 IVAR=LIST(ILN,2)/1000
  IF(IVAR,EQ,1) GO TO 30
C
C-----ERROR. FLOATING POINT VALUE TO BE SAVED FOR VARIABLE STORED
C IN STATE FILE AS AN INTEGER.
CALL PAGER(2)
WRITE(LP,1010) IFLD,(LIST(ILN,1),I=3,5)
RETURN
C
C-----FIELD TYPE IS CORRECT, UNPACK CODES FROM STATE FILE.
30 IS=1000*IVAR
  ITYP=(LIST(ILN,2)-IS)/10
  IS=LIST(ILN,2)-10*ITYP-IS
  IX=LIST(ILN,6)
C
C-----GENERATE HACS AUDIT DISPLAY FOR FIELD DEFINITION AND VALUE
C REQUESTED TO BE SAVED. NOTE THAT THE VALUE DISPLAYED MAY
C NOT BE SAVED IN THE STATE FILE.
CALL FCNV(IFLD,ILN,VAL,ITYP,ISRC)
C
C-----GENERATE WARNING MESSAGE IF VALUE TO BE SAVED EXCEEDS LIMITS.
IF(VAL,GE,FVAL(IX,2),AND,VAL,LE,FVAL(IX,3)) GO TO 40
IF(NPRRP,NE,0) GO TO 35
NPRRP=1
CALL FCNV(IFLD,ILN,VAL,ITYP,ISRC)
NPRRP=0
35 CONTINUE
CALL PAGER(5)
WRITE(LP,1030) IFLD,(LIST(ILN,1),I=3,5),FVAL(IX,2),FVAL(IX,3),
1 UNIT(1,ITYP)
C
C-----COMPARE SOURCE CODE OF VALUE TO BE SAVED TO SOURCE CODE OF
C VALUE ALREADY STORED IN STATE FILE. UPDATE ONLY IF NEW SOURCE
C CODE EXCEEDS PREVIOUS SOURCE CODE.
40 IF(ISRC,LT,IS) GO TO 50
  LIST(ILN,2)=LIST(ILN,2)+ISRC-IS
  FVAL(IX,1)=VAL
  RETURN
C
C-----WRITE NOTE THAT VALUE DID NOT HAVE HIGHER SOURCE CODE AND WAS
C NOT PLACED IN THE STATE FILE.
50 IF(NPRRP,NE,0) GO TO 55
  NPRRP=1
  CALL FCNV(IFLD,ILN,VAL,ITYP,ISRC)

```

```

NPRRP=0
55 CONTINUE
CALL PAGER(3)
WRITE(LP,1040) FVAL(IX,1),UNIT(1,ITYP),SOURC(IS+1)
RETURN

```

```

C 10000FORMAT (5X,26H*****ERROR - FIELD NUMBER ,I4,40H REQUESTED FOR SAVE
1 HAS NOT BEEN DEFINED/)
10100FORMAT (5X,51H*****ERROR - REAL SAVE REQUESTED FOR INTEGER FIELD ,
1 I4,1X,3A4/)
10300FORMAT (5X,30H*****WARNING - VALUE OF FIELD ,I4,1X,3A4/
1 10X,47HREQUESTED TO BE SAVED EXCEEDS NOMINAL LIMITS OF/
2 10X,G13.4,4H TO ,G13.4,2X,AB/10X,40HSUBSEQUENT CALCULATIONS MAY
3NOT BE VALID/)
10400FORMAT (5X,35H*****NOTE - VALUE IN STATE FILE OF ,G13.4,2X,AB/
1 10X,5HIS A ,AB,27H VALUE AND WAS NOT REPLACED/)
END
SUBROUTINE IRCL(IFLD,IVL,ISRC,IERR)

```

```

SUBROUTINE IRCL RECALLS THE VALUE (IVL) OF AN INTEGER FIELD,
DEFINED BY THE FIELD NUMBER IFLD, FROM THE HACS STATE FILE.
ERROR CONDITIONS WILL PRODUCE MESSAGES, AND CAUSE VALUES OF
IVL=0, IERR=1 AND ISRC=0 TO BE RETURNED. ARGUMENTS ISRC AND
IERR ARE INITIALIZED AND TESTED IN A CALLING PROGRAM TO
DETERMINE THE STATUS OF A GROUP OF DATA BASE RECALL OPERATIONS.

```

```

I      = INTEGER FORTRAN ARRAY INDEX
IERR   = ERROR CONDITION INDICATOR SET TO 1 IF ANY ERROR IS
        DETECTED IN PERFORMING THE REQUESTED RECALL
IFLD   = ARGUMENT, FIELD NUMBER FOR WHICH VALUE IS REQUESTED
ILN    = INDEX INTO STATE FILE TO OBTAIN DEFINITION OF FIELD
IS      = SOURCE CODE FOR VALUE CURRENTLY STORED IN STATE FILE
ISRC   = ARGUMENT, MINIMUM SOURCE CODE FOUND IN ONE OR MORE
        RECALL OPERATIONS GROUPED FOR INPUT TO A RATE MODEL
ITYP   = PRE-DEFINED TYPE OF PHYSICAL QUANTITY FOR FIELD IFLD
IVAR   = TYPE OF INTERNAL FIELD STORAGE FOR FIELD IFLD
        (0 FOR INTEGER, 1 FOR REAL)
IVL    = ARGUMENT, RETURNED AS VALUE OF FIELD IFLD, AS RECALLED
        FROM THE STATE FILE IN INTERNAL UNITS, OR 0 IF
IX      = INDEX STORED IN STATE FILE TO LINK FIELD DEFINITION
        TO FIELD VALUE ARRAYS
NFLD   = (NI+NF) GIVES THE TOTAL NUMBER OF FIELD DEFINITIONS
        ACTUALLY STORED IN THE STATE FILE

```

COMMON VARIABLES USED - IVAL,LIST,LP,NF,NI,SOURC,UNIT

SUBROUTINES REQUIRED - PAGER

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 29 JANUARY 1976

```

COMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
INTEGER UPTH
REAL MSG
DIMENSION STATE(2489)
EQUIVALENCE (STATE(1),MSG(1))

```

```

COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)

```

```

COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
COMMON/MODCN/MODEX(29),MODIO,LOCIO

```

```

COMMON/NAME/PTLST(30),SOURC(7)
INTEGER PTLST
LOGICAL YESNO,ENTR,INTEGR
LOGICAL QUEST

```

```

C-----LOOP THROUGH ALL FIELD DEFINITIONS IN STATE FILE TO LOCATE
C      FIELD NUMBER IFLD. NOTE THAT AT LEAST ONE FIELD (FOR CHEMICAL
C      RECOGNITION CODE) BY DEFINITION HAS BEEN PREVIOUSLY DEFINED.
C      THE VARIABLE ILN SAVES THE LOCATION IN THE STATE FILE
C      CORRESPONDING TO FIELD NUMBER IFLD, IF FOUND.
      NFLD=NI+NF
      DO 10 I=1,NFLD
      ILN=I
      IF(IFLD.EQ.LIST(I,1)) GO TO 20
10 CONTINUE

C-----ERROR. THE REQUESTED FIELD NUMBER DOES NOT EXIST IN THE HACs
C      STATE FILE. THIS CONDITION INDICATES EITHER AN ERROR IN
C      PROGRAM CODING OF CALLS TO SUBROUTINE IRCL, OR A MISSING
C      DEFAULT FILE SPECIFICATION TO DEFINE THE FIELD BEING REQUESTED.
      CALL PAGER(2)
      WRITE(LP,1000) IFLD
      GO TO 60

C-----VERIFY REQUEST FOR INTEGER VARIABLE WITH STORAGE MODE OF
C      VARIABLE IN STATE FILE
20 IVAR=LIST(ILN,2)/1000
   IF(IVAR.EQ.0) GO TO 30

C-----ERROR. INTEGER VALUE HAS BEEN REQUESTED FOR VARIABLE DEFINED
C      IN STATE FILE AS A FLOATING POINT VALUE.
      CALL PAGER(2)
      WRITE(LP,1010) IFLD,(LIST(ILN,I),I=3,5)
      GO TO 60

C-----FIELD TYPE IS INTEGER, UNPACK QUANTITY TYPE AND SOURCE CODE OF
C      STORED VALUE.
30 IS=1000*IVAR
   ITYP=(LIST(ILN,2)-IS)/10
   IS=LIST(ILN,2)-10*ITYP-IS

C-----UPDATE SUBROUTINE ARGUMENT TO TRACK LOWEST SOURCE CODE OF
C      FIELD VALUES RECALLED FOR USE IN EXECUTING A RATE MODEL.
   IF(IS.LT.ISRC) ISRC=IS

C-----IF FIELD VALUE HAS NOT BEEN DEFINED, SET ERROR FLAG AND USE
C      0 FOR STANDARD AUDIT.
   IF(IS.GT.0) GO TO 40
   IERR=1
   IVL=0
   GO TO 50

C-----INDEX INTO DATA ARRAY TO RETURN VALUE OF REQUESTED FIELD
40 IX=LIST(ILN,6)
   IVL=IVAL(IX,1)

C-----GENERATE HACs AUDIT DISPLAY FOR FIELD DEFINITION AND CURRENT
C      VALUE IN INTERNAL UNITS
50 IF(LOCIO.NE.2) GO TO 70

C-----SECTION FOR LOCIO=2, RECALL AND
C      DISPLAY FOR MODEL SUMMARY
51 IS=IS+1
   CALL PAGER(2)
   WRITE(LP,1020) IFLD,(LIST(ILN,I),I=3,5),IVL,UNIT(1,ITYP),SOURC(IS)
   IS=IS-1
   IF(LOCIO.NE.2) GO TO 80

C-----MIN/MAX TEST
53 CONTINUE
   IMN=IVAL(IX,2)
   IMX=IVAL(IX,3)
   IF(IVL.LT.IMN) GO TO 54
   IF(IVL.LE.IMX) GO TO 55
54 WRITE(LP,2030) IVL,IMN,IMX
20300FORMAT(24H WARNING, INPUT VALUE = ,IS/
1 29H NOT WITHIN NOMINAL RANGE OF ,IS,
2 4H TO ,IS)

```

```

55 CONTINUE
56 CONTINUE
57 IF(IS.LT.ISRC) ISRC=IS
   RETURN
70 IF(LOCIO.EQ.3) GO TO 71
   IF(IS.GT.1) GO TO 57
71 CONTINUE
72 WRITE(LP,2040) (LIST(ILN,I),I=3,5)
2040 FORMAT (25H ENTER INTEGER VALUE FOR ,3A4)
   IF(.NOT.ENTR(0)) GO TO 51
   IF(.NOT.QUEST(0)) GO TO 99
   CALL EXPLAIN(ILN)
   GO TO 72
99 CONTINUE
   IF(.NOT.INTEGR(IVL)) GO TO 72
   IVAL(IX,1)=IVL
   LIST(ILN,2)=LIST(ILN,2)-IS+5
   IS=5
   GO TO 53
80 WRITE(LP,2050)
2050 FORMAT (31H DO YOU WISH TO USE THIS VALUE?)
   IF(YESNO(0)) GO TO 53
   GO TO 72
C-----ERROR RETURN
60 IERR=1
   ISRC=0
   IVL=0
   RETURN
C
1000FORMAT (5X,26H*****ERROR - FIELD NUMBER ,I4,42H REQUESTED FOR RECA
      1LL HAS NOT BEEN DEFINED/)
10100FORMAT (5X,53H*****ERROR - INTEGER RECALL REQUESTED FOR REAL FIELD
      1 ,I4,1X,3A4/)
1020 FORMAT (5X,I4,1X,3A4,5H = ,I10,5X,A8,8H, IS A ,A8,7H VALUE/)
      END
      SUBROUTINE ISV(IFLD,IVL,ISRC)

```

SUBROUTINE ISV SAVES THE VALUE (IVL) OF AN INTEGER FIELD, GIVEN BY THE FIELD NUMBER IFLD, IN THE HACS STATE FILE DEPENDING ON THE SOURCE CODE. THE NEW VALUE IS SAVED IF ITS SOURCE CODE (ISRC) IS GREATER THAN THE SOURCE CODE OF THE VALUE ALREADY STORED IN THE STATE FILE. WHETHER OR NOT THE VALUE IS SAVED, THE ROUTINE PRODUCES AN OUTPUT AUDIT.

I = INTEGER FORTRAN ARRAY INDEX
IFLD = ARGUMENT, FIELD NUMBER FOR WHICH VALUE IS TO BE SAVED
ILN = INDEX INTO STATE FILE TO OBTAIN DEFINITION OF FIELD
IS = SOURCE CODE FOR VALUE CURRENTLY STORED IN STATE FILE
ISRC = ARGUMENT, SOURCE CODE ASSOCIATED WITH VALUE OF FIELD TO BE SAVED
ITYP = PRE-DEFINED TYPE OF PHYSICAL QUANTITY FOR FIELD IFLD
IVAR = TYPE OF INTERNAL FIELD STORAGE FOR FIELD IFLD (0 FOR INTEGER, 1 FOR REAL)
IVL = ARGUMENT, GIVES VALUE OF FIELD IFLD TO BE SAVED
IX = INDEX STORED IN STATE FILE TO LINK FIELD DEFINITION TO FIELD VALUE ARRAYS
NFLD = (NI+NF) GIVES THE TOTAL NUMBER OF FIELD DEFINITIONS ACTUALLY STORED IN THE STATE FILE

COMMON VARIABLES USED - IVAL,LIST,LP,NF,NI,SOURC,UNIT

SUBROUTINES REQUIRED - PAGER

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 3 FEBRUARY 1976

OCOMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,

```

1      NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
      INTEGER      UPTH
      REAL          MSG
      DIMENSION    STATE(2489)
      EQUIVALENCE (STATE(1),MSG(1))

C      COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)
C
C      COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
C
C      COMMON/NAME/PTLST(30),SOURC(7)
      INTEGER      PTLST

C
C-----LOOP THROUGH ALL FIELD DEFINITIONS IN STATE FILE TO LOCATE
C      FIELD NUMBER IFLD. NOTE THAT AT LEAST ONE FIELD (FOR CHEMICAL
C      RECOGNITION CODE) BY DEFINITION HAS BEEN PREVIOUSLY DEFINED.
C      THE VARIABLE ILN SAVES THE LOCATION IN THE STATE FILE
C      CORRESPONDING TO FIELD NUMBER IFLD, IF FOUND.
      NFLD=NI+NF
      DO 10 I=1,NFLD
      ILN=1
      IF(IFLD.EQ.LIST(I,1)) GO TO 20
10  CONTINUE

C-----ERROR. THE REQUESTED FIELD NUMBER DOES NOT EXIST IN THE HACS
C      STATE FILE. THIS CONDITION INDICATES EITHER AN ERROR IN
C      PROGRAM CODING OF CALLS TO SUBROUTINE ISV, OR A MISSING
C      DEFAULT FILE SPECIFICATION TO DEFINE THE FIELD BEING REQUESTED.
      CALL PAGER(2)
      WRITE(LP,1000) IFLD
      RETURN

C-----VERIFY REQUEST TO SAVE INTEGER VARIABLE WITH STORAGE MODE OF
C      VARIABLE IN STATE FILE.
20  IVAR=LIST(ILN,2)/1000
      IF(IVAR.EQ.0) GO TO 30

C-----ERROR. INTEGER VALUE TO BE SAVED FOR VARIABLE STORED IN
C      STATE FILE AS A REAL VARIABLE.
      CALL PAGER(2)
      WRITE(LP,1010) IFLD,(LIST(ILN,I),I=3,5)
10100  FORMAT (5X,51H****ERROR - INTEGER SAVE REQUESTED FOR REAL FIELD ,
1      I4,I1X,3A4/)
      RETURN

C-----FIELD TYPE IS CORRECT, UNPACK CODES FROM STATE FILE.
30  IS=1000*IVAR
      ITYP=(LIST(ILN,2)-IS)/10
      IS=LIST(ILN,2)-10*ITYP-IS
      IX=LIST(ILN,6)

C-----GENERATE HACS AUDIT DISPLAY FOR FIELD DEFINITION AND VALUE
C      REQUESTED TO BE SAVED. NOTE THAT THE VALUE DISPLAYED MAY
C      NOT BE SAVED IN THE STATE FILE.
      CALL PAGER(2)
      OWRITE(LP,1020) IFLD,(LIST(ILN,I),I=3,5),IVL,UNIT(1,ITYP),
1      SOURC(ISRC+1)

C-----GENERATE WARNING MESSAGE IF VALUE TO BE SAVED EXCEEDS LIMITS.
      IF(IVL.GE.IVAL(IX,2).AND.IVL.LE.IVAL(IX,3)) GO TO 40
      CALL PAGER(5)
      OWRITE(LP,1030) IFLD,(LIST(ILN,I),I=3,5),IVAL(IX,2),IVAL(IX,3),
1      UNIT(1,ITYP)

C-----COMPARE SOURCE CODE OF VALUE TO BE SAVED TO SOURCE CODE OF
C      VALUE ALREADY STORED IN STATE FILE. UPDATE ONLY IF NEW SOURCE
C      CODE EXCEEDS PREVIOUS SOURCE CODE.
40  IF(ISRC.LT.IS) GO TO 50
      LIST(ILN,2)=LIST(ILN,2)+ISRC-IS
      IVAL(IX,1)=IVL
      RETURN

```

```

C-----WRITE NOTE THAT VALUE DID NOT HAVE HIGHER SOURCE CODE AND WAS
C      NOT PLACED IN THE STATE FILE.
      50 CALL PAGER(3)
      WRITE(LP,1040) IVAL(IX,1),UNIT(1,ITYP),SOURC(IS+1)
      RETURN
C
10000FORMAT (5X,26H****ERROR - FIELD NUMBER ,I4,40H REQUESTED FOR SAVE
1      HAS NOT BEEN DEFINED/)
1020 FORMAT (5X,I4,1X,3A4,5H = ,I10,5X,A8,8H, IS A ,A8,7H VALUE/)
10300FORMAT (5X,30H****WARNING - VALUE OF FIELD ,I4,1X,3A4/
1      10X,47HREQUESTED TO BE SAVED EXCEEDS NOMINAL LIMITS OF/
2      10X,I10,3X,4H TO ,I10,5X,A8/10X,40HSUBSEQUENT CALCULATIONS MAY N
3      OT BE VALID/)
10400FORMAT (5X,35H****NOTE - VALUE IN STATE FILE OF ,I10,5X,A8/
1      10X,5HIS A ,A8,27H VALUE AND WAS NOT REPLACED/)
      END
      SUBROUTINE LABEL(X,ND,SCALE,NDIV)

      SUBROUTINE LABEL COMPUTES AN ARRAY OF AXIS LABELS, SCALE, TO
      BE USED IN PLOTTING VALUES OF THE VARIABLE X ALONG A SCALE
      DIVIDED INTO NDIV INTERVALS. THE NUMBER OF DATA POINTS IN
      THE ARRAY X IS GIVEN BY ND. THE DIMENSION OF THE ARRAY SCALE
      MUST BE SET IN THE CALLING PROGRAM AS NDIV+1 OR GREATER.
      SUBROUTINE LABEL ASSUMES THAT ND IS GIVEN ON INPUT AS 2
      OR GREATER. IF ND IS LESS THAN 2, ERRONEOUS RESULTS WILL
      BE RETURNED.

      THE ROUTINE EMPLOYS A ROUNDING TECHNIQUE USED BY K.M. WIIG
      AT ADL TO PRODUCE SMOOTH AXIS LABELS BASED SOLELY ON THE
      RANGE OF THE DATA TO BE PLOTTED, NOT PRE-DEFINED INTERVAL
      SIZES. THE RANGE OF THE PLOT SCALE PRODUCED IS ALWAYS EQUAL
      TO OR GREATER THAN THE RANGE OF DATA VALUES X.

      SEVERAL PARAMETERS ARE USED AND VALUES ARE PRE-SET IN DATA
      STATEMENTS TO SIMPLIFY ANY ADJUSTMENTS WHICH MAY BE DESIRED.
      IN PARTICULAR, THE PARAMETER FAC REPRESENTS A LIMITING RATIO
      OF THE MINIMUM VALUE OF X TO THE MAXIMUM VALUE. IF ALL X ARE
      POSITIVE, BUT XMIN IS LESS THAN FAC*XMAX, THE ORIGIN WILL
      AUTOMATICALLY BE INCLUDED IN THE PLOT SCALE.

      D      = GIVES THE NOMINAL RANGE OF THE PLOT SCALE (XMAX-XMIN)
      FAC      = MAXIMUM PROPORTION OF PLOT SCALE ALLOCATED TO INCLUDE
                  THE ORIGIN IF ALL X ARE POSITIVE
      I      = INTEGER LOOP INDEX, INTEGER VARIABLE FOR ROUNDING
                  OPERATION, AND MULTIPLE OF PROPORTIONAL SPACING
      IRND     = ARRAY OF DESIRED PROPORTIONAL SPACING MULTIPLES, WITH
                  DATA VALUES SET SUCH THAT IRND(I) IS GREATER THAN
                  OR EQUAL TO I FOR I LESS THAN OR EQUAL TO 20.
      J      = INTEGER LOOP INDEX OVER NUMBER OF INTERVALS NDIV
      K      = DUMMY SUBSCRIPT (=J+1) FOR INDEXING OUTPUT SCALE
                  FROM 2 TO NDIV+1
      LOGDEL   = INTEGER PART OF COMMON LOGARITHM
      ND      = ARGUMENT, NUMBER OF DATA POINTS GIVEN IN THE ARRAY X
                  (VALUE GIVEN ON INPUT MUST BE 2 OR GREATER)
      NDIV     = ARGUMENT, NUMBER OF DIVISIONS OF PLOT SCALE FOR WHICH
                  LABELS ARE DESIRED.
      RLIMB    = LIMIT VALUE USED TO CORRECT SMALL NUMBER ERROR
      RLIMB    = LIMIT VALUE USED TO CORRECT SMALL NUMBER ERROR
      RND      = VARIABLE ROUND-OFF FACTOR
      SCALE    = ARGUMENT, OUTPUT VALUES ARE DESIRED AXIS LABELS FROM
                  1 TO NDIV+1
      TENN     = TEN RAISED TO THE NTH POWER
      TENN1    = TEN RAISED TO THE (N-1) POWER
      TOL      = SET TOLERANCE USED TO CANCEL ROUND-OFF IF VALUE IS
                  VERY CLOSE TO, BUT ON WRONG SIDE, OF INTEGER VALUE
      X      = ARGUMENT, ARRAY OF N DATA POINTS
      XMAX     = MAXIMUM VALUE OF DATA POINTS X(I)
      XMIN     = MINIMUM VALUE OF DATA POINTS X(I), AND ADJUSTED
                  LOWER LIMIT OF PLOT SCALE

```

```

COMMON VARIABLES USED - NONE

SUBROUTINES REQUIRED - ABS, ALOG10, FLOAT

AUTHORS- K.M. WIIG, ARTHUR D. LITTLE, INC.,
          R.G. POTTS 35/309A ACORN PARK,
                   CAMBRIDGE, MASS., 02140
                   TEL. 617-864-5770 EXT. 2813

DATE - 16 OCTOBER 1975

      DIMENSION IRND(20),SCALE(1),X(1)
      ODATA      (IRND(I),I=1,20)/1,2,3,4,5,6,8,8,10,10,12,12,15,15,15,
1      DATA      FAC/0.20/,RLIMA/0.000000001/,RLIMB/0.0000000001/
      DATA      TOL/0.00001/

C-----SEARCH DATA ARRAY TO OBTAIN UPPER AND LOWER BOUNDS
      XMIN=X(1)
      XMAX=X(1)
      DO 10 I=2,ND
      IF(XMIN.GT.X(I)) XMIN=X(I)
      IF(XMAX.LT.X(I)) XMAX=X(I)
10  CONTINUE

C-----TEST FOR INCLUSION OF ORIGIN. IF SO, SET XMIN=0, AND JUMP
C      TO COMPUTE AXIS LABELS. OTHERWISE SET UP ROUND-OFF FACTORS
C      FOR XMIN.
      IF(XMIN) 30,60,20
20  IF(XMIN.GE.FAC*XMAX) GO TO 40
      XMIN=0.0
      GO TO 60
30  RND=TOL-1.0
      GO TO 50
40  RND=TOL

C-----COMPUTE LOWER BOUND OF PLOT SCALE. OBTAIN NOMINAL RANGE D,
C      RESET IF SMALL NUMBER ERROR OCCURS.
50  D=XMAX-XMIN
      IF(D.LE.0.0) D=RLIMA
C-----OBTAIN INTEGER PART OF COMMON LOGARITHM, AND COMPUTE 10**N-1.
      LOGDEL=ALOG10(D)
      TENN=10.**LOGDEL
      TENN1=TENN/10.
C-----ROUND-OFF LOWER LIMIT FOR PLOT SCALE
      XMIN=XMIN/TENN1
      I=XMIN+RND
      XMIN=I*TENN1

C-----COMPUTE RANGE OF PLOT SCALE. GET NOMINAL RANGE AND ADJUST
C      FOR VERY SMALL VALUES
60  D=ABS(XMAX-XMIN)
      IF(D.LE.0.0) D=RLIMB
C-----OBTAIN INTEGER PART OF COMMON LOGARITHM, ADJUST IF D IS LESS
C      THAN 1.0, AND COMPUTE 10**N
      LOGDEL=ALOG10(D)
      IF(D.LT.1.0) LOGDEL=LOGDEL-1
      TENN=10.**LOGDEL
C-----COMPUTE MULTIPLE FOR DESIRED RANGE FACTOR
      I=(20./FLOAT(NDIV))*(D/TENN)+1.0-TOL
C-----REMOVE COMMENT ON STATEMENT BELOW TO USE ARRAY IRND TO
C      ELIMINATE UNDESIRABLE MULTIPLES OF I
      IF(I.LT.20) I=IRND(I)
C-----COMPUTE COORDINATES FOR PLOT AXIS LABELING.
      SCALE(1)=XMIN
      TENN=FLOAT(I)*TENN/20.
      DO 70 J=1,NDIV
      K=J+1
70  SCALE(K)=XMIN+TENN*FLOAT(J)
      RETURN

```

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 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
 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
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525

ՀԱՅԿԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ ՎՃԱՌԱԿԱՆ ԱՄՔԱՆՈՒՄ

ՀԱՅԿԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ ՎԵՐԱԴԱՐԱՆ

1. **Содержание**
 2. **Введение**
 3. **Глава 1. Общие сведения о компании**
 4. **Глава 2. Описание продукции**
 5. **Глава 3. Анализ рынка**
 6. **Глава 4. Финансовый анализ**
 7. **Глава 5. Заключение**
 8. **Приложение**
 9. **Список литературы**
 10. **Индекс**

**ՀԱՅԿԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ՎՃԱՌԱՅԱԿԱՆ ՄԵԼՆԵՐ**

**ՀԱՅԿԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ՎՃԱՌԱՅԱԿԱՆ ՄԵԼՈՒՆԵՐ**

[illegible]

THE UNIVERSITY OF CHICAGO

[illegible]

QUESTION

QUESTION

```

C      FOR ISW = 1, 2, OR 3
C      ISYS = OPTION SELECTED BY USER FOR OUTPUT SYSTEM OF UNITS
C              (VALUES ARE 0=1, 2, 3, 4)
C      ITYP = FIELD QUANTITY TYPE RETRIEVED FROM STATE FILE
C      IVAR = FIELD STORAGE TYPE RETRIEVED FROM STATE FILE
C              (0 = INTEGER, 1 = REAL)
C      IX = INDEX RETRIEVED FROM STATE FILE, POINTS TO ARRAY
C              LOCATIONS GIVING FIELD VALUES
C      J = DUMMY ARRAY INDEX
C      JSRC = HAS VALUE OF ISRC+1, IN RANGE 1 TO 7, TO INDEX DATA
C              ARRAY OF OUTPUT LABELS.
C      JSYS = HAS VALUE OF ISYS-1 TO INDEX ARRAY OF CONVERSION
C              FACTORS. SYSTEM 1 GIVES INTERNAL UNITS, AND
C              CONVERSION FACTORS ARE NOT STORED.
C      NFLD = TOTAL NUMBER OF FIELDS DEFINED IN STATE FILE.
C      TAG = TEMPORARY STORAGE OF FIELD UNIT OUTPUT LABEL

COMMON VARIABLES USED - CONV,FVAL,IVAL,LBL,LIST,LP,LSTCN,
                      MSG,NF,NI,SOURC,STCON,UNIT

SUBROUTINES REQUIRED - PAGER

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
        35/309A ACORN PARK,
        CAMBRIDGE, MASS., 02140
        TEL. 617-864-5770 EXT. 2813

DATE - 22 JANUARY 1976

C      OCOMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1      NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
C      INTEGER      UPTH
C      REAL          MSG

C      OCOMMON/CNTRL/EOFF,ICD,IDFLT,LBL(4),LSTCN(3,3),MODEL(15),NOP,
1      STCON,SVCON
C      INTEGER      EOFF,STCON,SVCON
C      REAL          LBL

C      COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)
C      COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
C      COMMON/NAME/PTLST(30),SOURC(7)
C      INTEGER      PTLST

C      INTEGER      CP
C      DIMENSION    FAC(3),IN(3,4)
C      DATA        CP/62/
C      DATA        (FAC(I),I=1,3)/1.0,1.8,1.0/
C      DATA        (IN(1,I),I=1,4)/4HINIT,4HIALI,4HZATI,4HON /
C      DATA        (IN(2,I),I=1,4)/4HUSER,4H INP,4HUT ,4H /
C      DATA        (IN(3,I),I=1,4)/4HASSE,4HSSME,4HNT R,4HUN /

C      C-----TEST FOR REQUESTED LIST OPTION AND RETURN IF NOT SELECTED
C      IF(LSTCN(ISW,1).EQ.0) RETURN

C      C-----WRITE HEADER FOR OPTION SELECTED AND FILE TYPE
C      IF(ISW.NE.1) CALL PAGER(0)
C      CALL PAGER(3)
C      WRITE(LP,1000) (IN(ISW,I),I=1,4)
C      CALL PAGER(1)
C      WRITE(LP,1010) LBL(STCON)

C      C-----IF STATE FILE IS UNDEFINED OR EMPTY, SKIP OUTPUT WITH MESSAGE
C      AND RETURN
C      IF(STCON.LE.1) GO TO 10
C      NFLD=NF+NI
C      IF(NFLD.GT.1) GO TO 20
10  CALL PAGER(3)

```

```

        WRITE(LP,1020)
        RETURN
C-----RETRIEVE FILE DISPLAY OPTIONS, OVERRIDE DEFAULT FOR UNIT
C      SELECTION
20  ISYS=LSTCN(ISW,2)
    IF(ISYS.EQ.0) ISYS=1
    JSYS=ISYS-1
    IPN=LSTCN(ISW,3)
C-----WRITE STATE FILE HEADER
C      CALL PAGER(2)
        WRITE(LP,1030) MSG
        IF(IPN.EQ.1) WRITE(CP,1040) MSG
        CALL PAGER(3)
        WRITE(LP,1050)
        CALL PAGER(3)
        WRITE(LP,1060)
C-----LOOP ON NUMBER OF FIELDS DEFINED IN FILE
C      DO 60 I=1,NFLD
C-----UNCODE DATA FROM STATE FILE FOR FIELD IN LIST POSITION I
C      IFLD=LIST(I,1)
        IVAR=LIST(I,2)/1000
        ISRC=1000*IVAR
        ITYP=(LIST(I,2)-ISRC)/10
        ISRC=LIST(I,2)-10*ITYP-ISRC
        JSRC=ISRC+1
        IX=LIST(I,6)
        TAG=UNIT(ISYS,ITYP)
C-----IF FIELD IS REAL, APPLY OUTPUT CONVERSION AND DISPLAY
C      IF(IVAR.EQ.0) GO TO 40
        A=FVAL(IX,1)
        B=FVAL(IX,2)
        C=FVAL(IX,3)
        IF(JSYS.LE.0) GO TO 30
        D=CONV(JSYS,ITYP)
C-----BRANCH FOR TEMPERATURE CONVERSION EQUATION
C      IF(ITYP.EQ.6) GO TO 25
        A=A/D
        B=B/D
        C=C/D
        GO TO 30
25  A=D+A*FAC(JSYS)
        B=D+B*FAC(JSYS)
        C=D+C*FAC(JSYS)
30  CALL PAGER(1)
        OWRITE(LP,1070) IFLD,A,TAG,IVAR,ISRC,SOURC(JSRC),B,C,
1      ITYP,(LIST(I,J),J=3,5)
        OIF(IPN.EQ.1) WRITE(CP,1080) IFLD,A,TAG,IVAR,B,C,
1      ITYP,(LIST(I,J),J=3,5)
        GO TO 60
C-----FOR INTEGER FIELDS, DISPLAY ALL FIELDS IN SAME FORMAT EXCEPT
C      FOR FIELD 1001, CHEMICAL RECOGNITION CODE
40  IF(IFLD.NE.1001) GO TO 50
        CALL PAGER(1)
        OWRITE(LP,1090) IFLD,IVAL(IX,1),TAG,IVAR,ISRC,
1      SOURC(JSRC),ITYP,(LIST(I,J),J=3,5)
        IF(IPN.EQ.1) WRITE(CP,1100) IFLD,IVAL(IX,1),TAG
        GO TO 60
C
50  CALL PAGER(1)
        OWRITE(LP,1110) IFLD,IVAL(IX,1),TAG,IVAR,ISRC,
1      SOURC(JSRC),IVAL(IX,2),IVAL(IX,3),ITYP,(LIST(I,J),J=3,5)
        OIF(IPN.EQ.1) WRITE(CP,1120) IFLD,IVAL(IX,1),TAG,IVAR,
1      IVAL(IX,2),IVAL(IX,3),ITYP,(LIST(I,J),J=3,5)
C
60  CONTINUE
C

```


CALL OVERLAY(6HUIMABS,NOV,0,6HRECALL)

RETURN

END

SUBROUTINE PAGER(LINES)

SUBROUTINE COMPARES THE NUMBER OF OUTPUT LINES TO BE WRITTEN ON UNIT LP, SPECIFIED BY THE ARGUMENT LINES, TO THE NUMBER OF LINES REMAINING ON THE PAGE, LNPG-LNCT, AND WRITES A HEADER AT THE TOP OF A NEW PAGE WHEN INSUFFICIENT SPACE IS LEFT TO WRITE A BLOCK OF LENGTH LINES. A NEW PAGE MAY BE FORCED BY CALLING THE ROUTINE WITH LINES SET TO ZFRO AS THE ARGUMENT. THE HEADER LINE PRODUCED CONSISTS OF AN 80 CHARACTER TITLE, DATE AND PAGE NUMBER. THE PAGE NUMBER IS AUTOMATICALLY UPDATED BY THIS ROUTINE.

LINES = NUMBER OF OUTPUT LINES TO BE WRITTEN IMMEDIATELY FOLLOWING CALL TO PAGER
TIME = TIME OF DAY OBTAINED FROM SYSTEM LIBRARY MACRO AS 24-HOUR CLOCK READING IN A8 FORMAT AS HH/MM/SS

COMMON VARIABLES USED - DATE, LNCT, LNPG, LP, NPG, TITLE

SUBROUTINE REQUIRED - TMDY

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 27 MAY 1975

COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)

-----TEST FOR TOP OF FORM COMMAND, I.E., CALL PAGER(0)
IF(LINES.EQ.0) GO TO 10

-----INCREMENT CUMULATIVE LINE COUNT WITH NUMBER OF LINES TO BE WRITTEN FOLLOWING CALL AND COMPARE TO PAGE LIMIT. RETURN IF LESS THAN LIMIT.

LNCT=LNCT+LINES
IF(LNCT.LE.LNPG) RETURN

-----PAGE LIMIT REACHED OR EXCEEDED. INCREMENT PAGE COUNTER, WRITE HEADER AND SET LINE COUNT TO NUMBER OF LINES IN HEADER PLUS OUTPUT LINES FOLLOWING.

10 NPG=NPG+1
CALL TIME(TIM)
WRITE(LP,1000) DTE,TIM,NPG,TITLE
LNCT=LINES+6
RETURN

10000FORMAT (46H1HAZARD ASSESSMENT COMPUTER SYSTEM (HACS) DATE,A10,
1 1X,4HTIME,A9,2X,4HPAGE,I3//1X,9A8,A7/1X,79(1H#)//)

END

SUBROUTINE PLTLP(PTITL,X,Y,N,XTITL,YTITL,II,DIV,XTITL1)

SUBROUTINE PLTLP PRODUCES A ONE PAGE 40 BY 80 LINE PRINTER PLOT OF THE DATA POINTS (X(J),Y(J)), J=1,N. THE ARGUMENTS TO THE ROUTINE PROVIDE FOR A PLOT TITLE, IDENTIFICATION ALONG EACH AXIS AND AN OPTION TO PRODUCE A DOUBLE SET OF LABELS FOR THE X-AXIS. ALL LABEL ARRAYS ARE STANDARDIZED AT 48 CHARACTERS (6A8) EACH AND MAY BE USED FOR LABELS, DIMENSIONS OR OTHER INFORMATION. THE Y-AXIS LABEL IS SPLIT INTO THREE PARTS OF 16 CHARACTERS EACH TO KEEP THE FINISHED PLOT WITHIN AN 8.5 BY 11 INCH LIMIT. SMOOTH AXIS LABELING IS PERFORMED BY SUBROUTINE LABEL.

EXACTLY N DATA POINTS ARE PLOTTED AS SPECIFIED BY THE ARGUMENTS X AND Y. NO DATA SMOOTHING OR INTERPOLATION IS PERFORMED BY THIS ROUTINE. THERE ARE NO RESTRICTIONS ON THE FORM OF THE FUNCTION REPRESENTED BY X AND Y.

```

C      BUFF      = 81 CHARACTER PRINT BUFFER WITH A PLOT SYMBOL SET AT
C      EACH RELATIVE X POSITION FOR ANY DATA POINT HAVING
C      A VALUE OF Y CORRESPONDING TO NY
C      DIV      = ARGUMENT, A FACTOR WHICH GIVES THE RELATIONSHIP
C      BETWEEN THE TWO X AXES TO BE PRINTED IF II=1.
C      THE LABELS GENERATED FOR THE FIRST X AXIS ARE
C      DIVIDED BY DIV TO PRODUCE THE LABELS FOR THE
C      SECOND AXIS.
C      DX      = COMPUTED GRID SIZE ALONG X-AXIS
C      DY      = COMPUTED GRID SIZE ALONG Y-AXIS
C      IBLNK    = INTEGER DATA WORD CONTAINING BLANKS (A4) USED TO
C      INITIALIZE PRINT BUFFER
C      II      = ARGUMENT, SWITCH USED TO SELECT (II=1) OR SUPPRESS
C      (II=0) PRINTING OF SECOND X AXIS
C      ISW      = ALTERNATING VARIABLE (+1,-1) USED FOR FORMAT CONTROL
C      ON ALTERNATING LINES
C      J      = INTEGER INDEX, GENERAL
C      JY      = INTEGER INDEX DECREMENTED TO DISPLAY Y AXIS LABELS
C      FROM TOP OF PAGE DOWN
C      KX      = COMPUTED INTEGER POSITION IN PRINT BUFFER WHICH
C      CORRESPONDS TO X COORDINATE OF DATA POINT
C      KY      = COMPUTED INTEGER POSITION ALONG VERTICAL SCALE
C      WHICH CORRESPONDS TO Y COORDINATE OF DATA POINT
C      N      = ARGUMENT, NUMBER OF DATA POINTS TO BE DISPLAYED
C      NY      = INTEGER INDEX CORRESPONDING TO VERTICAL SCALE, VARIES
C      FROM 41 TO 1 FROM THE TOP DOWN
C      NYY      = DUMMY INDEX LOOP FROM 1 TO 41 USED TO OBTAIN NY
C      PLOT     = SYMBOL USED FOR DATA POINT DISPLAY
C      PTITL    = ARGUMENT, 48 CHARACTER PLOT TITLE
C      X      = ARGUMENT, ARRAY OF VALUES OF X COORDINATES, N POINTS
C      XMAX     = UPPER LIMIT OF X AXIS (IS GENERALLY EQUAL TO OR
C      GREATER THAN THE MAXIMUM OF X(J))
C      XMIN     = LOWER LIMIT OF X AXIS (IS GENERALLY EQUAL TO OR
C      LESS THAN THE MINIMUM OF X(J))
C      XSCL     = ARRAY OF VALUES FOR LABELS ALONG FIRST ABSCISSA
C      XSCL1    = ARRAY OF VALUES FOR LABELS ALONG SECOND ABSCISSA
C      XTITL    = ARGUMENT, 48 CHARACTER TITLE FOR FIRST ABSCISSA
C      XTITL1   = ARGUMENT, 48 CHARACTER TITLE FOR SECOND ABSCISSA
C      Y      = ARGUMENT, ARRAY OF VALUES OF Y COORDINATES, N POINTS
C      YMAX     = UPPER LIMIT OF Y AXIS (SEE XMAX)
C      YMIN     = LOWER LIMIT OF Y AXIS (SEE XMIN)
C      YSCL     = ARRAY OF VALUES FOR LABELS ON ORDINATE
C      YTITL    = ARGUMENT, 48 CHARACTER TITLE FOR ORDINATE

```

COMMON VARIABLES USED - LP

SUBROUTINES REQUIRED - LABEL,PAGER

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
 35/309A ACORN PARK,
 CAMBRIDGE, MASS., 02140
 TEL. 617-864-5770 EXT. 2813

DATE - 16 OCTOBER 1975

REVISION - 12 MARCH 1980 (SCALE TO 80 CHAR PLOTS)

COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)

```

DIMENSION BUFF(61),PTITL(6),X(1),XSCL(7),XSCL1(7),XTITL(6),
1 XTITL1(6),Y(1),YSCL(21),YTITL(6)
INTEGER BUFF,PLOT
EQUIVALENCE (XMIN,XSCL(1)),(XMAX,XSCL(7))
EQUIVALENCE (YMIN,YSCL(1)),(YMAX,YSCL(21))
DATA IBLNK/4H /,PLOT/4H0 /

```

```

C-----SET UP X AND Y AXES FOR EVEN LABELS AND COMPUTE GRID SIZES
CALL LABEL(X,N,XSCL,6)
DX=(XMAX-XMIN)/60.
CALL LABEL(Y,N,YSCL,20)
DY=(YMAX-YMIN)/40.

```

```

C-----WRITE PLOT TITLE AND AXIS LINE
      CALL PAGER(0)
      WRITE(LP,1000) (YTITL(I),I=1,4),(PTITL(I),I=1,6),YTITL(5),YTITL(6)
      WRITE(LP,1010)
C-----SET UP AND LOOP ON Y SCALE.      CLEAR PRINT BUFFER EACH TIME
      JY=22
      ISW=-1
      DO 70 NYY=1,41
      NY=42-NYY
      ISW=-ISW
      DO 10 J=1,61
10  BUFF(J)=IBLNK
C-----LOOP THROUGH ALL DATA POINTS.  IF Y COORDINATE MATCHES
C      CURRENT PRINT LINE, SET PLOT SYMBOL IN POSITION CORRESPONDING
C      TO X COORDINATE.
      DO 20 J=1,N
      KY=(Y(J)-YMIN)/DY+1.5
      IF(KY.NE.NY) GO TO 20
      KX=(X(J)-XMIN)/DX+1.5
      BUFF(KX)=PLOT
20  CONTINUE
C-----SELECT TYPE OF PRINT LINE
      IF(ISW.LT.0) GO TO 30
      JY=JY-1
      WRITE(LP,1020) YSCL(JY),BUFF
      GO TO 70
30  CONTINUE
      WRITE(LP,1030) BUFF
      GO TO 70
70  CONTINUE
C-----CLOSE OFF PLOT WITH SINGLE OR DOUBLE X-AXIS LABELS
      WRITE(LP,1040)
      WRITE(LP,1060) XSCL,XTITL
      IF(II.EQ.0) RETURN
      DO 80 J=1,7
80  XSCL1(J)=XSCL(J)/DIV
      WRITE(LP,1060) XSCL1,XTITL1
      RETURN
C-----BLANK SUBSTITUTED FOR +
1000 FORMAT (///1X,2A8/1X,2A8,9X,6A8/1X,2A8/4X,7(9X,1H) )
C-----BLANK SUBSTITUTED FOR *
1010 FORMAT (12X,63(1H) )
C-----TWO BLANKS SUBSTITUTED FOR *+
1020 FORMAT (1X,1PE9.2,3H +*,61A1,2H )
C-----BLANK SUBSTITUTED FOR *
1030 FORMAT (12X,1H*,61A1,1H )
1040 FORMAT (12X,63(1H*))
1060 FORMAT (4X,7(9X,1H+)/7X,7(1X,1PE9.2)/19X,6A8)
C
      END
      SUBROUTINE SEGLOD(NSEG)

      INTERFACE ROUTINE TO SEGMENT LOAD FUNCTION WHERE NSEG GIVES
      THE NUMBER OF THE SEGMENT TO BE LOADED.  ALL CALLS TO THE
      SEGMENT LOAD FUNCTION ARE CODED WITHIN EACH OVERLAY REQUIRING
      SEGMENTS, AND THE SELECTION IS USUALLY BASED ON THE RATE MODEL
      INDEX MODNO.  THE SEGMENT INDEX LIST SGLST ORIGINALLY DEFINED
      FOR THIS PURPOSE IS NOT GENERALLY USED.

      COMMON VARIABLES USED - NONE

      SUBROUTINES REQUIRED - UFSEG

      AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
      35/309A ACORN PARK,
      CAMBRIDGE, MASS., 02140
      TEL. 617-864-5770 EXT. 2813

```

```

C      DATE - 16 OCTOBER 1975
C
C      IF(NSEG.LE.0) RETURN
C      CALL OVERLAY(6HUIHABS,8,NSEG,6HRECALL)
C      RETURN
C      END
C      SUBROUTINE TOXIC(C,IFLAG,AM,XCONC)
C
C      THIS SUBROUTINE CONVERTS CONCENTRATIONS OF A VAPOR IN AIR FROM UNI
C      OF GM/CM**3. TO UNITS OF PPM OR MOLE PERCENT
C
C      C****INPUTS
C
C      C      THE CONCENTRATION IN GM/CM**3. WHOSE UNITS ARE TO BE CONVE
C      IFLAG  FLAG INDICATING WHETHER DESIRED OUTPUT IS TO BE IN UNITS O
C      PPM OR MOLE PERCENT. IFLAG=0 FOR MOLE PERCENT,IFLAG=1 FO
C      AM      MOLECULAR WEIGHT OF THE CHEMICAL
C
C      C****OUTPUTS
C
C      XCONC  THE CONCENTRATION DESIRED IN EITHER UNITS OF MOLE PERCENT
C
C      DENA=0.0012894
C      DENV=AM/22414.
C      XCONC=1./((1.+((AM/28.9)*(DENA/C)*(1.-(C/DENV))))
C      IF(IFLAG=0) 20,20,10
C 10  XCONC=XCONC*1000000.
C      GO TO 30
C 20  XCONC=XCONC*100.
C 30  RETURN
C      END
C      SUBROUTINE TRACE(ISW,NOV,NSEG)
C
C      SUBROUTINE TRACE PROVIDES LINE PRINTER MESSAGES FOR DIAGNOSTIC
C      TESTING OF HACS OVERLAY FUNCTION. SUBROUTINE CALLS ARE CODED
C      AT THE BEGINNING AND END OF EACH OVERLAY TO BRANCH TO THIS
C      ROUTINE TO SELECT AN AUDIT MESSAGE FOR THE STATUS OF EACH
C      OVERLAY AND SEGMENT. A RETURN STATEMENT HAS BEEN INSERTED AS
C      THE FIRST EXECUTABLE STATEMENT TO OVER-RIDE THIS DIAGNOSTIC
C      FUNCTION FOR USER RUNS.
C
C      ISW    = MESSAGE CONTROL SWITCH, 0 TO FORCE STARTING MESSAGE,
C              1 TO SELECT ENDING MESSAGE
C      NOV    = OVERLAY NUMBER SET IN CALLING PROGRAM
C      NSEG   = SEGMENT NUMBER SET IN CALLING PROGRAM
C
C      COMMON VARIABLES USED - LP
C
C      SUBROUTINES REQUIRED - PAGER
C
C      AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
C              35/309A ACORN PARK,
C              CAMBRIDGE, MASS., 02140
C              TEL. 617-864-5770 EXT. 2813
C
C      DATE - 2 DECEMBER 1975
C
C      COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)
C
C      RETURN
C      CALL PAGER(1)
C      IF(ISW.EQ.1) GO TO 10
C      WRITE(LP,1000) NOV,NSEG
C      RETURN
C 10  WRITE(LP,1010) NOV,NSEG
C      RETURN
C
C 10000FORMAT (5X,42H****OVERLAY TRACE, NOW EXECUTING OVERLAY ,I2,
C      1 9H SEGMENT ,I2)
C 10100FORMAT (5X,37H****OVERLAY TRACE, FINISHED OVERLAY ,I2,
C      1 9H SEGMENT ,I2)
C      END

```

```

        LOGICAL FUNCTION ENTR(J)
C        READS TERMINAL INPUT INTO
C        BUFFER FOR CHARACTER SCAN
        OCOMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
        1  NUM,OUT,PTR,SPLST(14),TYP,YES
        INTEGER BLANK,BUFF,CHAR,DEC,EXP
        INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
        EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
        EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
C-----CLEAR EOF INDICATOR
        IN=5
        IF(EOF(IN)) 10,10
C-----INITIALIZE BUFFER
        10 DO 20 I=1,80
        20 BUFF(I)=BLANK
        READ(IN,1000) BUFF
        PTR=0
C-----SEE IF INPUT FOUND
        30 IF(NEXT(0)) 30,40,50
        40 ENTR=.FALSE.
        RETURN
        50 ENTR=.TRUE.
        PTR=0
        RETURN
        1000 FORMAT (80A1)
        END
        LOGICAL FUNCTION FLTPT(VALUE)
C-----RETURNS .TRUE. WITH VALID REAL NUMBER AS VALUE
C-----FALSE, OTHERWISE
        OCOMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
        1  NUM,OUT,PTR,SPLST(14),TYP,YES
        INTEGER BLANK,BUFF,CHAR,DEC,EXP
        INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
        EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
        EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
        LOGICAL INTEGR
C-----INITIALIZE FOR VALID RETURN
        FLTPT=.TRUE.
C-----SKIP TO NEXT NON-BLANK INPUT CHARACTER
        10 IF(NEXT(0)) 10,150,20
C-----INITIALIZE
        20 TEMP=0.0
        VALUE=0.0
        ISIGN=1
C-----CHECK FOR OPTIONAL SIGN
        IF(CHAR.EQ.PLUS) GO TO 30
        IF(CHAR.NE.MINUS) GO TO 40
        ISIGN=-1
C-----NEED NEXT INPUT, MAY BE DECIMAL POINT OR DIGIT
        30 IF(NEXT(0)) 150,150,40
        40 IF(TYP.NE.1) GO TO 50
C-----PROCESS DIGITS PRECEEDING DECIMAL POINT
        TEMP=10.*TEMP+NUM
        IF(NEXT(0)) 130,130,40
C-----CHECK FOR DECIMAL POINT OR EXPONENT
        50 IF(CHAR.EQ.EXP) GO TO 75
        IF(CHAR.NE.DEC) GO TO 140
        PT=1.0
C-----PROCESS DIGITS FOLLOWING DECIMAL POINT
        60 IF(NEXT(0)) 130,130,65
        65 IF(TYP.NE.1) GO TO 70
        PT=PT/10.
        TEMP=TEMP+PT*NUM
        GO TO 60
C-----TEST AND PROCESS OPTIONAL EXPONENT
        70 IF(CHAR.NE.EXP) GO TO 140
        75 IF(.NOT.INTEGR(IEXP)) GO TO 150
        IF(IEXP) 80,130,90
        80 EXBS=0.1
        IEXP=-IEXP
        GO TO 100
        90 EXBS=10.

```

```

100 DO 110 J=1,IEXP
110 TEMP=TEMP*EXBS
C-----STORE RESULT AND RETURN
130 VALUE=TEMP*ISIGN
RETURN
C-----BAD CHARACTER ERROR, ADVANCE TO BLANK OR EOR
140 IF(NEXT(0)) 150,150,140
C-----ERROR ON BLANK OR EOR
150 FLTPT=.FALSE.
OUT=6
WRITE(OUT,1000)
RETURN
1000 FORMAT (29H WHAT? (DECIMAL SYNTAX ERROR))
END
LOGICAL FUNCTION INTEGR(IRESLT)
C-----RETURNS .TRUE. WITH VALID INTEGER AS IRESLT
C-----RETURNS .FALSE. FOR INVALID INTEGER WITH IRESLT = 0
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
C-----INITIALIZE FOR VALID RETURN
INTEGR=.TRUE.
C-----SKIP TO NEXT NON-BLANK INPUT CHARACTER
10 IF(NEXT(0)) 10,70,20
C-----INITIALIZE
20 IRESLT=0
ITEMP=0
ISIGN=1
C-----CHECK FOR OPTIONAL SIGN
IF(CHAR,EQ,PLUS) GO TO 30
IF(CHAR,NE,MINUS) GO TO 40
ISIGN=-1
C-----NEED NEXT INPUT, MUST BE VALID DIGIT
30 IF(NEXT(0)) 70,70,40
40 IF(TYP,NE,1) GO TO 60
C-----ADD DIGIT TO INTEGER RESULT
ITEMP=10*ITEMP+NUM
C-----GET NEXT INPUT
IF(NEXT(0)) 50,50,40
C-----HAVE RESULT, STOPPED ON BLANK OR EOR
50 IRESLT=ISIGN*ITEMP
RETURN
C-----BAD CHARACTER ERROR, ADVANCE TO BLANK OR EOR
60 IF(NEXT(0)) 70,70,60
C-----ERROR ON BLANK OR EOR
70 INTEGR=.FALSE.
OUT=6
WRITE(OUT,1000)
RETURN
1000 FORMAT (29H WHAT? (INTEGER SYNTAX ERROR))
END
LOGICAL FUNCTION NAME(TAG)
C-----READS UP TO 10 CHARACTER NAME IN A1 FORMAT
C-----RETURNS RESULT IN A10 FORMAT, BLANK FILL RIGHT
C-----RETURNS .FALSE. IF NO INPUT, .TRUE. OTHERWISE
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
DATA MASK ( 1)/77000000000000000000B/
DATA MASK ( 2)/00770000000000000000B/
DATA MASK ( 3)/00007700000000000000B/
DATA MASK ( 4)/00000077000000000000B/
DATA MASK ( 5)/00000000770000000000B/
DATA MASK ( 6)/00000000007700000000B/

```

```

DATA MASK ( 7)/000000000000077000000B/
DATA MASK ( 8)/000000000000000770000B/
DATA MASK ( 9)/000000000000000007700B/
DATA MASK (10)/00000000000000000077B/
C-----SKIP TO NON-BLANK
10 IF(NEXT(0)) 10,70,20
C-----INITIALIZE PACKING
20 TAG=0B
I=0
ISHFT=0
C-----PACK CHARACTER I
30 I=I+1
ITEMP=SHIFT(CHAR,ISHFT)
ITEMP=ITEMP.AND.MASK(I)
TAG=TAG.OR.ITEMP
ISHFT=ISHFT+6
C-----TESTS FOR END
IF(CHAR.EQ.BLANK) GO TO 50
IF(I.GE.10) GO TO 80
C-----GET NEXT CHAR TO BE PACKED
IF(NEXT(0)) 40,40,30
40 CHAR=BLANK
GO TO 30
C-----LAST CHAR PACKED WAS BLANK
C CONTINUE WITH BLANK FILL TO END
50 IF(I.LT.10) GO TO 30
60 NAME=.TRUE.
RETURN
C-----RETURN FOR EMPTY INPUT RECORD
70 TAG=BLANK
NAME=.FALSE.
RETURN
C-----AFTER TENTH CHARACTER IS PACKED,
C MOVE TO EOR OR FIRST BLANK CHARACTER
C IN INPUT.
80 IF(NEXT(0)) 60,60,80
RETURN
END
FUNCTION NEXT(J)
C-----SCANS INPUT FOR NEXT CHARACTER.
C VALUE = -1 FOR BLANK
C VALUE = 0 FOR EOR
C VALUE = +1 FOR NON-BLANK AND ALSO GIVES
C CHAR, NUM, TYP
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
C-----INCREMENT POINTER AND TEST FOR EOR
PTR=PTR+1
IF(PTR.GT.80) GO TO 20
C-----TEST FOR BLANK
IF(BUFF(PTR).EQ.BLANK) GO TO 30
C-----HAVE NON BLANK CHARACTER, SET NUMBER AND
C TYPE FOR SPECIAL CHARACTERS, TYPE 6
C FOR ALL OTHERS.
NEXT=1
CHAR=BUFF(PTR)
DO 10 I=1,14
IF(CHAR.NE.SPLST(I)) GO TO 10
NUM=I-1
TYP=1
IF(I.GT.10) TYP=I-9
RETURN
10 CONTINUE
TYP=6
RETURN
C-----EOR RETURN
20 NEXT=0
GO TO 40

```

```

C-----BLANK RETURN
30 NEXT=-1
40 CHAR=BLANK
RETURN
END
LOGICAL FUNCTION YESNO(J)
C-----RETURNS .TRUE. FOR YES, .FALSE. FOR NO
LOGICAL ENTR
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
C-----DISPLAY PROMPT
10 CONTINUE
OUT=6
C-----READ INPUT
IF(.NOT.ENTR(0)) GO TO 50
C-----SKIP TO FIRST NON-BLANK, OR EOR
20 IF(NEXT(0)) 20,50,30
C-----RETURN IF Y OR N
30 IF(CHAR.NE.YES) GO TO 40
YESNO=.TRUE.
RETURN
40 IF(CHAR.NE.NO) GO TO 50
YESNO=.FALSE.
RETURN
50 CONTINUE
OUT=6
WRITE(OUT,1010)
WRITE(OUT,1000)
GO TO 10
C
1000 FORMAT (23H ENTER YES OR NO (Y/N):)
1010 FORMAT (6H WHAT?)
END
BLOCK DATA
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))
DATA BLANK/1H /,IN/5/,NO/1HN/,OUT/6/,YES/1HY/
ODATA (SPLST(I),I=1,14)/1H0,1H1,1H2,1H3,1H4,
1 1H5,1H6,1H7,1H8,1H9,1H+,1H-,1H.,1HE/
END
SUBROUTINE EXPLAIN(NMSG)
DIMENSION MSG(71),TXT(70)
INTEGER ONE,TERM,THR,TWO,TXT
EQUIVALENCE (MSG(2),TXT(1))
ODATA MSG(1)/10H(5X, "/,ONE/1H1/,TERM/2H"/,THR/1H3/,
1 TWO/1H2/
CALL READMS(11,TXT,69,NMSG)
NW=LENGTH(11)
IF(TXT(1).EQ.ONE) GO TO 10
IF(TXT(1).EQ.TWO) GO TO 30
NW=NW+1
TXT(NW)=TERM
IF(TXT(1).EQ.THR) GO TO 20
WRITE(6,MSG)
GO TO 40
10 WRITE(6,1000)
GO TO 40
20 TXT(1)=MSG(1)
WRITE(6,TXT)
30 WRITE(6,1010)
40 WRITE(6,1020)
RETURN
10000FORMAT (5X,60HREFER TO HACS USER'S REFERENCE MANUAL FOR FIELD DESC
1RIPTION.)

```

10100FORMAT (5X,58HREFER TO CHRIS MANUAL II FOR CHEMICAL PROPERTY DATA
1ITEMS.)

1020 FORMAT (5X)

END
LOGICAL FUNCTION QUEST(J)
COMMON/INTER/BLANK,BUFF(80),CHAR,IN,NO,
1 NUM,OUT,PTR,SPLST(14),TYP,YES
INTEGER BLANK,BUFF,CHAR,DEC,EXP
INTEGER OUT,PLUS,PTR,SPLST,TYP,YES
EQUIVALENCE (DEC,SPLST(13)),(EXP,SPLST(14))
EQUIVALENCE (MINUS,SPLST(12)),(PLUS,SPLST(11))

C INTEGER QUERY
C DATA QUERY/10H?

QUEST=,FALSE.
IF(CHAR.EQ.QUERY) QUEST=,TRUE.
RETURN
END
SUBROUTINE SNMSG(I)

C
C SUBROUTINE PRINTS SCENARIO MESSAGES. NOTE THAT THIS FUNCTION
C COULD LIKELY BE COMBINED WITH SUBROUTINE EXPLAIN.

C
C DIMENSION MSG(71),TXT(70)
C INTEGER TERM,TXT
C EQUIVALENCE (MSG(2),TXT(1))
C DATA MSG(1)/10H(9X, "/,TERM/2H")/
C CALL READMS(12,TXT,69,1)
C NW=LENGTH(12)
C NW=NW+1
C TXT(NW)=TERM
C WRITE(6,MSG)
C RETURN
C END
C SUBROUTINE INIT(CODE,I,J,K)

C
C SUBROUTINE INIT INITIALIZES THE CODING ROUTINES TO STORE NEW
C CODES, OR TO READ PREVIOUSLY STORED CODES, IN THE INTEGER
C ARRAY CODE. THE ARRAY CODE, MUST BE DIMENSIONED IN THE
C CALLING PROGRAM TO BE OF LENGTH J OR GREATER. THE CHARACTER-
C ISTICS OF THE STORED NUMERIC CODES ARE SPECIFIED BY THE
C REMAINING ARGUMENTS -

C
C I = MAXIMUM NUMBER OF BITS IN EACH WORD OF THE ARRAY CODE
C WHICH CAN BE USED FOR STORAGE OF CODED VALUES.
C J = MAXIMUM NUMBER OF WORDS IN ARRAY CODE WHICH ARE
C USED FOR STORAGE OF CODED VALUES.
C K = DEFINES THE STORAGE REQUIRED FOR A SINGLE CODED VALUE
C TO BE FIXED LENGTH AT K BITS PER CODE. THIS DETER-
C MINES THE ALLOWED INTEGER MAGNITUDE OF EACH CODED
C VALUE TO BE GREATER THAN OR EQUAL TO ZERO, AND LESS
C THAN 2**K.

C
C ON RETURN, THE ERROR FLAG IERR IN COMMON IS ZERO IF NO ERRORS
C WERE ENCOUNTERED. ERROR CONDITIONS WILL CAUSE IERR TO BE SET
C TO 1,2,7,8 OR 9 ON RETURN, AND CONTROL VARIABLES IN COMMON
C TO BE SET FOR SINGLE BIT, SINGLE WORD CODE STORAGE.

C
C SUBROUTINE INIT CONTAINS A SINGLE INTERNAL PARAMETER, MXWRD,
C WHICH DEFINES THE MAXIMUM ALLOWED UNSIGNED INTEGER WORD
C LENGTH IN BITS AND IS INSTALLATION DEPENDENT. FOR A NORMAL
C 16-BIT WORD LENGTH, MXWRD SHOULD BE SET TO 15. FOR USE WITH
C DOUBLE PRECISION (TWO-WORD) INTEGERS, MXWRD CAN BE SET TO 31
C FOR A 16-BIT WORD LENGTH IF INTEGER SPECIFICATIONS ARE ALSO
C MODIFIED IN THESE ROUTINES. FOR USE ON THE CDC CYBERNET NET-
C WORK, INTEGER ARITHMETIC IS LIMITED TO PARTIAL WORDS, SO MXWRD
C IS SET TO 47 OUT OF 60 BITS AVAILABLE IN THE FULL WORD.

C
C SUBROUTINE INIT MUST BE CALLED ONCE AND ONLY ONCE FOR EACH
C CODED ARRAY PRIOR TO ALL CALLS USING THE ROUTINES SET, RSET
C OR ITST WITH THE CODED ARRAY. NOTE THAT INIT WILL CLEAR THE

```

C      CONTENTS OF THE REFERENCED CODED ARRAY.  INIT MUST BE CALLED
CCCCC IMMEDIATELY BEFORE EACH CALL TO THE BULK TRANSFER ROUTINES
C      PACK AND UNPK.
C
C      COMMON VARIABLES USED - CDLN,CPW,IERR,L,MAXN,MAXV
C
C      SUBROUTINES REQUIRED - NONE
C
C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
C      INTEGER      CPW,CDLN,SHFT
C
C      INTEGER      CODE(1)
C      DATA        MXWRD/47/
C
C-----TEST LENGTH OF WORD TO BE USED FOR CODE STORAGE.  CANNOT BE
C      LESS THAN 1 OR EXCEED MAXIMUM UNSIGNED INTEGER WORD LENGTH.
C      IF(I.LE.0) GO TO 20
C      IF(I.GT.MXWRD) GO TO 30
C
C-----TEST NUMBER OF WORDS TO BE USED FOR CODE STORAGE.  CANNOT BE
C      LESS THAN 1.  UPPER LIMIT IS NOT TESTED SINCE THIS IS
C      CONTROLLED BY USER DIMENSION IN CALLING PROGRAM.
C      IF(J.LE.0) GO TO 40
C
C-----TEST NUMBER OF BITS TO BE USED FOR SINGLE CODE.  CANNOT BE
C      LESS THAN 1 OR EXCEED SPECIFIED LENGTH OF CODE WORD.
C      IF(K.LE.0) GO TO 50
C      IF(K.GT.I) GO TO 60
C
C-----NORMAL RETURN.  COMPUTE NUMBER OF CODES TO BE STORED PER
C      WORD (CPW), INITIALIZE ALL CODE WORDS TO ZERO, AND SET NORMAL
C      ERROR RETURN.  COMPUTE TOTAL NUMBER OF CODES WHICH CAN BE
C      STORED (MAXN), MOVE CODE LENGTH K TO COMMON VARIABLE CDLN,
C      AND COMPUTE MAXIMUM ALLOWED CODE VALUE (MAXV).
C      IERR=0
C      CPW=I/K
C      DO 10 L=1,J
C10  CODE(L)=0
C      MAXN=CPW*J
C      CDLN=K
C      MAXV=2**CDLN-1
C      RETURN
C
C-----ERROR RETURNS.  SET VALUE OF ERROR SWITCH IN COMMON AND
C      DEFAULT TO CODE DEFINITION USING SINGLE WORD CONTAINING CODES
C      ONE BIT IN LENGTH.
C      20 IERR=1
C      GO TO 70
C      30 IERR=2
C      GO TO 70
C      40 IERR=7
C      GO TO 70
C      50 IERR=8
C      GO TO 70
C      60 IERR=9
C      70 CPW=MXWRD
C      CODE(1)=0
C      MAXN=MXWRD
C      CDLN=1
C      MAXV=1
C      RETURN
C      END
C      LOGICAL FUNCTION ECHK(N)
C
C      LOGICAL FUNCTION ECHK (FOR ERROR CHECK) TESTS THE REQUESTED
C      CODE POSITION SPECIFIED BY THE ARGUMENT N.  IF THE POSITION
C      IS NOT WITHIN THE ALLOWED NUMBER OF CODED VALUES (1 TO MAXN),
C      THE ERROR INDICATOR IERR IN COMMON IS SET TO 3 OR 4 AND THE
C      FUNCTION RETURNS A VALUE OF .TRUE.  ALL OTHER VARIABLES IN
C      COMMON ARE UNCHANGED.

```

IF THE SPECIFIED CODE POSITION, N, IS VALID, THE ERROR CHECK FUNCTION RETURNS A VALUE OF .FALSE. AND SETS VARIABLES IN COMMON TO ACCESS THE VALUE OF THE NTH CODE PACKED IN AN ARRAY. GIVEN N, THE LOCATION OF THE CODED VALUE IS DETERMINED BY THE NUMBER OF CODED VALUES PER STORAGE WORD (CPW) AND THE LENGTH OF EACH CODE (CDLN). BOTH CPW AND CDLN ARE DETERMINED ON INITIALIZATION IN SUBROUTINE INIT. FOR ACCESSING THE REQUESTED CODE THE FUNCTION RETURNS L AND SHFT. THE VALUE OF L IS THE SUBSCRIPT INDEX TO THE WORD OF THE PACKED ARRAY CONTAINING THE POSITION FOR THE CODED VALUE. SHFT IS AN INTEGER MULTIPLIER OR DIVISOR WHICH WILL MOVE A CODED VALUE OF LENGTH CDLN TO OR FROM ITS POSITION IN WORD L FROM OR TO THE LOW ORDER NUMERIC POSITION.

COMMON VARIABLES USED - CDLN,CPW,IERR,L,MAXN,SHFT,ITMP

SUBROUTINES REQUIRED - NONE

COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
 INTEGER CPW,CDLN,SHFT
 EQUIVALENCE (I,IPOS,ITMP)

-----TEST REQUESTED CODE POSITION. MUST LIE WITHIN DEFINED
 BOUNDARY OF CODE WORD STRUCTURE.
 IF(N.LE.0) GO TO 10
 IF(N.GT.MAXN) GO TO 20

-----NORMAL RETURN. SET ERROR CODE AND FUNCTION VALUE.
 IERR=C
 ECHK=.FALSE.

-----COMPUTE WORD ADDRESS (L) WITHIN CODE LIST ARRAY, AND POSITION
 ADDRESS (IPOS) WITHIN WORD L FOR CODE LOCATION N.
 I=N-1
 L=I/CPW
 IPOS=I-L*CPW
 L=L+1

-----COMPUTE SHIFT FACTOR TO ACCESS CODE N IN POSITION IPOS OF
 WORD L.
 I=CDLN*IPOS
 SHFT=2**I
 RETURN

-----ERROR RETURNS.
 10 IERR=3
 GO TO 30
 20 IERR=4
 30 ECHK=.TRUE.
 RETURN
 END
 FUNCTION ITST(CODE,N)

FUNCTION ITST RETURNS THE INTEGER VALUE OF CODE N STORED IN A PACKED ARRAY CODE. IF N IS NOT WITHIN THE RANGE OF THE PACKED CODES, A VALUE OF ZERO IS RETURNED FOR ITST AND IERR IS SET TO 3 OR 4. IF N IS VALID, THE VALUE OF ITST IS OBTAINED FROM THE PACKED CODE IN POSITION N IN THE RANGE 0 TO MAXV, AND IERR IS RETURNED AS ZERO. DEFINITION OF THE PACKED CODE STRUCTURE IS OBTAINED FROM THE MOST RECENT CALL TO SUBROUTINE INIT.

COMMON VARIABLES USED - ITMP,L,MAXV,SHFT

SUBROUTINES REQUIRED - ECHK

COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
 INTEGER CPW,SHFT,CDLN

```

C      INTEGER      CODE(1)
      LOGICAL      ECHK
C
C-----INITIALIZE FUNCTION VALUE AND RETURN IF N IS INVALID.
      ITST=0
      IF(ECHK(N)) RETURN
C-----MOVE CODED VALUE IN WORD L TO LOW ORDER POSITION OF ITMP.
      ITMP=CODE(L)/SHFT
C-----OBTAIN CODED VALUE BY REMOVING ANY BITS REMAINING IN HIGHER
      ORDER POSITIONS.
      ITST=ITMP.AND.MAXV
      RETURN
      END
      SUBROUTINE SUMRY
C
C      COMMON VARIABLES USED - ICD,LP,MODEL,PTLST,SNCOD
C      SUBROUTINES REQUIRED - INIT,ITST,PAGER,SNMSG,SPRNT,YESNO
C      DATE - 23 JANUARY 1981
C
C      OCOMMON/CNTRL/EOFF,ICD,IDFLT,LBL(4),LSTCN(3,3),MODEL(15),NOP,
1      STCON,SVCON
      INTEGER      EOFF,STCON,SVCON
      REAL      LBL
C
C      COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
C
C      COMMON/NAME/PTLST(30),SOURCE(7)
      INTEGER      PTLST
C
C      COMMON/PXFER/BUFF(15),K1,SNCOD
      INTEGER      BUFF,SNCOD
C
C      DIMENSION ITXT(5),IVAL(5),SCLST(28)
      INTEGER      SCLST
      LOGICAL      YESNO
C
C      ODATA (SCLST(I),I=1,28)/3HA B,3HA C,5HA B C,5HA D E,7HA D F G,
1      9HA D E F G,3HA H,5HA I J,7HA H I J,5HA K L,7HA K M N,
2      9HA K L M N,3HA O,3HA P,5HA P Q,7HA P R S,9HA P Q R S,
3      3HA T,5HA T U,5HA V W,9HA T U V W,3HA X,5HA X Y,1HZ,2HII,
4      2HRR,4HRR C,2HSS/
C
C      CALL PAGER(2)
      WRITE(LP,1000)
      IF(.NOT.YESNO(0)) RETURN
      DO 10 I=1,5
      IVAL(I)=0
10      ITXT(I)=0
      CALL PAGER(2)
      WRITE(LP,1010)
      CALL PAGER(2)
      WRITE(LP,1020)
      IF(.NOT.YESNO(0)) GO TO 20
      IVAL(1)=1
      CALL PAGER(2)
      WRITE(LP,1030)
      IF(YESNO(0)) ITXT(1)=1
20      CALL PAGER(2)
      WRITE(LP,1040)
      IF(.NOT.YESNO(0)) GO TO 30
      IVAL(2)=1
      CALL PAGER(2)
      WRITE(LP,1030)

```

```

      IF(YESNO(0)) ITXT(2)=1
30  CALL PAGER(2)
    WRITE(LP,1050)
    IF(.NOT.YESNO(0)) GO TO 40
    IVAL(3)=1
    CALL PAGER(2)
    WRITE(LP,1030)
    IF(YESNO(0)) ITXT(3)=1
40  CALL PAGER(2)
    WRITE(LP,1060)
    IF(.NOT.YESNO(0)) GO TO 50
    IVAL(4)=1
    CALL PAGER(2)
    WRITE(LP,1030)
    IF(YESNO(0)) ITXT(4)=1
50  CALL PAGER(2)
    WRITE(LP,1080)
    IF(.NOT.YESNO(0)) GO TO 70
    IVAL(5)=1
    CALL PAGER(2)
    WRITE(LP,1030)
    IF(YESNO(0)) ITXT(5)=1
C-----START REPORT
70  CALL PAGER(0)
    CALL PAGER(7)
    WRITE(LP,1090) ICD
C-----SCENARIO SUMMARY
    IF(IVAL(1).EQ.0) GO TO 90
    CALL PAGER(4)
    WRITE(LP,1100)
    CALL INIT(ITMP,28,1,1)
    DO 80 I=1,28
      ITMP=ITST(SNCD,I)
      IF(ITMP.EQ.0) GO TO 80
      CALL PAGER(3)
      WRITE(LP,1110) SCLST(I)
      IF(ITXT(1).EQ.1) CALL SNMSG(I)
80  CONTINUE
C-----MODEL SUMMARY
90  IF(IVAL(2).EQ.0) GO TO 110
    CALL PAGER(4)
    WRITE(LP,1120)
    DO 100 I=1,15
      J=MODEL(I)
      IF(J.GE.30) GO TO 100
      IF(J.LE.0) GO TO 100
      CALL PAGER(3)
      WRITE(LP,1130) PTLST(J)
      IF(ITXT(2).EQ.1) CALL MODEXP(J)
100 CONTINUE
C-----USER INPUT SUMMARY
110 IF(IVAL(3).EQ.0) GO TO 120
    CALL PAGER(4)
    WRITE(LP,1150)
    CALL SPRNT(5,5,ITXT(3))
C-----COMPUTED VALUE SUMMARY
120 IF(IVAL(4).EQ.0) GO TO 130
    CALL PAGER(4)
    WRITE(LP,1160)
    CALL SPRNT(2,6,ITXT(4))
C-----CHEMICAL PROPERTY VALUES
130 IF(IVAL(5).EQ.0) GO TO 150
    CALL PAGER(4)
    WRITE(LP,1180)
    CALL SPRNT(2,3,ITXT(5))
C-----END
150 CALL PAGER(5)
    WRITE(LP,1190)
    RETURN
C
1000 FORMAT (/49H DO YOU WANT TO PRINT A SUMMARY OF THESE RESULTS?)
1010 FORMAT (/38H WHICH OF THE FOLLOWING DO YOU WANT?)

```



```

      5 ICNT=ICNT+1
      IX=LIST(ILN,6)
      IF(IVAR.EQ.0) GO TO 10
C-----DISPLAY REAL FIELD VALUE
      VAL=FVAL(IX,1)
      CALL FCNV(IFLD,ILN,VAL,ITYP,IS)
      GO TO 20
C-----DISPLAY INTEGER VALUE
      10 CALL PAGER(2)
      OWRITE(LP,1010) IFLD,(LIST(ILN,I),I=3,5),IVAL(IX,1),
      1 UNIT(1,ITYP),SOURC(IS+1)
C-----TEST FOR TEXT AUDIT
      20 IF(IT.EQ.1) CALL EXPLAIN(ILN)
      30 CONTINUE
C-----ADD MESSAGE IF NO OUTPUT IS PRINTED
      IF(ICNT.GT.0) RETURN
      CALL PAGER(1)
      WRITE(LP,1020)
      RETURN
C
      1010 FORMAT (5X,I4,1X,3A4,5H = ,I10,5X,A8,8H, IS A ,A8,7H VALUE/)
      1020 FORMAT (10X,13HNOT PERTINENT)
      END
      SUBROUTINE MODEXP(NMSG)
      DIMENSION MSG(192),TXT(191)
      INTEGER TERM,TXT
      EQUIVALENCE (MSG(2),TXT(1))
      DATA LP/6/,MSG(1)/10H(9X, ' /,TERM/3H'//
      CALL READMS(13,TXT,190,NMSG)
      NW=1+LENGTH(13)
      TXT(NW)=TERM
      WRITE(LP,MSG)
      RETURN
      END
      OVERLAY(1,0)
      PROGRAM INPUT

```

PROGRAM INPUT PERFORMS THE FUNCTIONS OF INTERPRETING ALL USER DATA, ACCESSING/LOADING/UPDATING THE DEFAULT FILE AND ACCESSING THE HACS PHYSICAL PROPERTY FILE. THE BASIC STRUCTURE OF ALL HACS DATA DECKS IS AS FOLLOWS -

1. RUN CONTROL OPTION CARD
2. RUN TITLE CARD
3. ASSESSMENT PATH CARD
4. CHEMICAL RECOGNITION CODE CARD
5. ONE OR MORE FIELD DATA CARDS
6. DATA DECK SEPARATOR CARD (9999) OR END OF FILE CARD

WHERE THE SPECIFIC PROCESSING OF EACH TYPE OF DATA DECK IS DETERMINED BY THE RUN CONTROL OPTION WHICH HAS BEEN SELECTED.

THE CONTENT AND FORMAT OF FIELD DATA CARDS PROCESSED BY HACS ARE DEFINED BELOW -

IFLD = INTEGER FIELD NUMBER (UP TO FOUR DIGITS IN C.C. 1 TO 4). VALUE MUST BE POSITIVE, AND A VALUE OF 9999 TERMINATES FIELD DATA INPUT.

FVL = FIELD VALUE, ENTERED IN C.C. 5 TO 19 IN FIXED POINT FORMAT. ALL INTEGERS MUST BE RIGHT JUSTIFIED. REAL FIELD VALUES WITHOUT EXPONENT MAY BE LOCATED ANYWHERE IN FIELD IF DECIMAL POINT IS GIVEN. EXPONENTIAL NOTATION MAY BE USED, BUT EXPONENT FIELD MUST BE RIGHT JUSTIFIED.

TAG = UNIT LABEL, UP TO EIGHT CHARACTERS LEFT JUSTIFIED IN C.C. 20 TO 27. APPLIES TO FIELD VALUE FVL, AND ALSO TO MIN AND MAX VALUES IF GIVEN. IF THE UNIT FIELD IS BLANK, DIMENSIONS OF INPUT VALUES ARE ASSUMED TO BE CONSISTENT WITH PRE-DEFINED UNITS USED IN HACS INTERNAL COMPUTATIONS.

IR = FLAG FOR INTEGER (0) OR REAL (1) FIELD VALUES IN C.C. 28. INPUT VALUES ARE ALLOWED ONLY DURING CREATION OF DEFAULT FILE, AND IN ORDER TO CHANGE THIS VALUE THE ENTIRE DEFAULT FILE MUST BE RE-LOADED. NOTE

THAT CALLS TO SAVE AND RECALL ROUTINES ARE CODED
DEPENDING ON IR, SO CHANGES IN THIS VALUE MAY ALSO
REQUIRE CODE CHANGES. (REFER TO ERROR CONDITIONS
IN SAVE/RECALL.)

MM = INDICATOR IN C.C. 29 USED TO CONTROL INPUT OF RANGE
VALUES AS FOLLOWS -
0 = NEITHER VALUE ENTERED
1 = ONLY MINIMUM VALUE ENTERED IN C.C. 30 TO 44
2 = ONLY MAXIMUM VALUE ENTERED IN C.C. 45 TO 59
3 = BOTH MINIMUM AND MAXIMUM VALUES ENTERED
WHEN LOADING THE DEFAULT FILE, BOTH MIN AND MAX
VALUES MUST BE GIVEN AND MM MUST BE 3.

FMN = MINIMUM FIELD VALUE, ENTERED IN C.C. 30 TO 44 IN
FIXED POINT FORMAT (RIGHT JUSTIFIED). VALUE IS
READ IF MM IS 1 OR 3, AND CONVERTED DEPENDING ON
INPUT OF UNIT LABEL, TAG. SEE ALSO FVL ABOVE.

FMX = MAXIMUM FIELD VALUE, ENTERED IN C.C. 45 TO 59 IN
FIXED POINT FORMAT (RIGHT JUSTIFIED). VALUE IS
READ IF MM IS 2 OR 3, AND CONVERTED DEPENDING ON
INPUT OF UNIT LABEL, TAG. SEE ALSO FVL ABOVE.

ITP = INTEGER SPECIFICATION IN C.C. 60-61, RIGHT JUSTIFIED,
FOR TYPE OF PHYSICAL QUANTITY (USED TO CONTROL UNIT
CONVERSION AND DISPLAY)

IN = UP TO 12 CHARACTER FIELD NAME, LEFT JUSTIFIED IN C.C.
62 TO 73, USED TO IDENTIFY OUTPUT

--- = C.C. 74 TO 80 ARE NOT USED FOR INPUT DATA FIELDS
EXCEPT THAT THE CORRESPONDENCE TO PROPERTY FILE
FIELDS HAS BEEN GIVEN ON DEFAULT CARDS. ANY OTHER
SEQUENCE OR IDENTIFICATION INFORMATION MAY BE
ENTERED IF DESIRED.

THE CONTENT OF THE HACS DEFAULT FILE IS DEFINED BY THE
SEQUENCE IN WHICH DATA CARDS ARE READ DURING THE CREATE
DEFAULT FILE OPTION. CONSECUTIVE ARRAY POSITIONS ARE LOADED
SEQUENTIALLY. THE ONLY FIXED DEFINITION IS THAT LIST(1,1)
IS RESERVED FOR FIELD 1001, CHEMICAL RECOGNITION CODE.

CNT = COUNT OF FIELD DATA CARDS READ BY SUBROUTINE BASIC,
USED TO ABORT HACS RUN IF DEFAULT FILE IS TO RE
CREATED OR UPDATED BUT NO CARDS ARE READ

CR = INTEGER FORTRAN UNIT NUMBER FOR CARD READER

FBLNK = DATA WORD SET TO ALL BLANKS USED TO INITIALIZE LABELS

I = DUMMY SUBSCRIPT, ARRAY INDEX

IARG = ARRAY FOR READING USER OPTIONS FOR HACS STATE FILE
DISPLAYS

IBLNK = DATA WORD SET TO ALL BLANKS FOR TESTING CHEMICAL
RECOGNITION CODE

IERR = ERROR INDICATOR (=1), OTHERWISE 0. USED TO ABORT
RUN IF ERROR OCCURS WHILE READING DEFAULT DATA

IFLD = FIELD NUMBER ON RECOGNITION CODE CARD, MUST BE 1001

IN = ARRAY USED FOR READING LABELS ON USER DATA CARDS

ISW = ERROR STATUS INDICATOR RETURNED BY PATH CHECK ROUTINE

IX = TEMPORARY VARIABLE USED TO UPDATE SOURCE CODE OF
RECOGNITION CODE IN STATE FILE

J = DUMMY SUBSCRIPT OR ARRAY INDEX

K = SUBSCRIPT USED IN STORING FILE DISPLAY OPTIONS

L = SUM OF FILE DISPLAY OPTION VALUES ENTERED BY USER,
USED TO GENERATE AUDIT ONLY IF ONE OR MORE OPTIONS
WERE SELECTED

OPLST = ARRAY CONTAINING THE FIRST FOUR CHARACTERS OF EACH
VALID RUN CONTROL OPTION

COMMON VARIABLES USED - EOF,ICD,IDFLT,IPRAC,IVAL,LBL,LIST,
LP,LSTCN,MNF,MNI,MODEL,MSG,MSYS,NF,NI,
NOP,PTLST,SAVE,STATE,STCON,SVCON,TITLE,
UPTH

SUBROUTINES REQUIRED - ACCESS,BASIC,IFEOF,LSTFL,PAGER,PROP,
PTHCK,RNTIO,TRACE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,

C
C
C
C

CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 12 FEBRUARY 1976

OCOMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
INTEGER UPTH
REAL MSG
DIMENSION STATE(2489)
EQUIVALENCE (STATE(1),MSG(1))

C

OCOMMON/CNTRL/EOFF,ICD,IDFLT,LBL(4),LSTCN(3,3),MODEL(15),NOP,
1 STCON,SVCON
INTEGER EOFF,STCON,SVCON
REAL LBL

C

COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)

C

COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)

C

COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP

C

COMMON/NAME/PTLST(30),SOURC(7)
INTEGER PTLST

C

COMMON/PXFER/BUFF(15),K1,SNCOD
INTEGER BUFF,SNCOD
DIMENSION IARG(6),IN(4),OPLST(5)
INTEGER TAGX
LOGICAL ENTR,NAME
LOGICAL QUEST
LOGICAL YESNO
LOGICAL INTEGR
INTEGER CNT,CR,OPLST
INTEGER SCLST(28)
INTEGER EXIT
DATA EXIT/4HEXIT/
ODATA (SCLST(I),I=1,28)/3HA B,3HA C,5HA B C,5HA D E,7HA D F G,
1 9HA D E F G,3HA H,5HA I J,7HA H I J,5HA K L,7HA K M N,
2 9HA K L M N,3HA O,3HA P,5HA P Q,7HA P R S,9HA P Q R S,
3 3HA T,5HA T U,5HA V W,9HA T U V W,3HA X,5HA X Y,1HZ,2HII,
4 2HRR,4HRR C,2HSS/
DATA CR/60,FBLNK/10H /,IBLNK/4H /
ODATA (OPLST(I),I=1,5)/8HRUN ,8HRERUN ,8HCONTINUE,
1 8HEND ,8H /

C

C-----INITIALIZE
CALL TRACE(0,1,0)

C

C-----RETURN HERE TO READ NEXT CONTROL CARD. TERMINATE ON END OF FILE

10 NOP=0
WRITE(LP,2000)
20000FORMAT(32H ENTER RUN REQUEST, OPTIONS ARE ,
1 24H(RUN/RERUN/CONTINUE/END))
IF(.NOT.ENTR(0)) GO TO 10
IF(.NOT.NAME(TAGX)) GO TO 10
IF(TAGX.EQ.OPLST(4)) GO TO 430

C

C-----SEARCH LIST OF FIRST FOUR CHARACTERS OF CONTROL OPTIONS
11 DO 20 NOP=1,3
IF(TAGX.EQ.OPLST(NOP)) GO TO 21
20 CONTINUE

C

C-----TERMINATE ON INVALID CONTROL CARD
WRITE(LP,2010)
2010 FORMAT(26H WHAT? (RUN REQUEST ERROR))
GO TO 10
21 DO 22 I=1,6
22 IARG(I)=0

```

      GO TO 60
C
30 CALL PAGER(3)
   WRITE(LP,1020)
   STOP
C
C-----READ TITLE CARD AND STORE FOR OUTPUT PAGE HEADING
40 CONTINUE
   GO TO 60
C
C-----TERMINATE ON UNEXPECTED END OF FILE
50 CALL PAGER(3)
   WRITE(LP,1030)
   GO TO 30
C
C-----BRANCH ON OPERATION CONTROL FLAG TO INITIALIZE STATE FILE
60 GO TO(70,90,120,130,70),NOP
C
70 IF(STCON.EQ.2) GO TO 150
80 IF(SVCON.EQ.2) GO TO 100
   CALL ACCESS(STATE,IDFLT,0,STCON)
   GO TO 150
C
90 IF(SVCON.GT.1) GO TO 100
   CALL ACCESS(SAVE,IDFLT,0,SVCON)
100 DO 110 I=1,2489
110 STATE(I)=SAVE(I)
   STCON=SVCON
   GO TO 150
C
120 IF(STCON.LE.1) GO TO 80
   GO TO 150
C
130 STCON=1
   SVCON=1
   NF=0
   NI=0
   DO 140 I=1,10
140 MSG(I)=FBLNK
C
C-----START NEW PAGE AND WRITE FILE INITIALIZATION MESSAGE
150 CALL PAGER(0)
   CALL PAGER(5)
   WRITE(LP,1040) LBL(STCON),MSG
C
C-----DECODE FILE LIST OPTIONS AND STORE IN LSTCN FOR USE IN
C      SUBROUTINE LSTFL.
C
      IERR=0
      L=0
      K=0
      DO 190 I=1,3
      DO 160 J=2,3
      K=K+1
      L=L+IARG(K)
160 LSTCN(I,J)=IARG(K)
C
C-----OVERRIDE ANY USER INPUT ERRORS, BUT SET ERROR FLAG TO PRODUCE
C      AUDIT MESSAGE.
C
      IF(LSTCN(I,2).LE.MSYS) GO TO 180
170 IERR=1
      LSTCN(I,1)=0
      GO TO 190
180 IF(LSTCN(I,2).LT.0) GO TO 170
      IF(LSTCN(I,3).LT.0) GO TO 170
      IF(LSTCN(I,3).GT.1) GO TO 170
      LSTCN(I,1)=LSTCN(I,2)+LSTCN(I,3)
      IF(LSTCN(I,1).GT.0) LSTCN(I,1)=1
190 CONTINUE
C
C-----DISPLAY STATE FILE AFTER INITIALIZATION IF OPTION SELECTED,
C      THEN PUT USER TITLE IN STATE FILE AND WRITE HEADER FOR LISTING
C      OF USER INPUT DATA

```

```

      CALL LSTFL(1)
      DO 200 I=1,10
200  MSG(I)=TITLE(I)
      GO TO 500
C-----
C-----READ PATH CODE INPUT CARD.  TERMINATE IF INSTEAD GET END OF
C-----FILE, OTHERWISE AUDIT INPUT THEN VALIDATE.  RUN IS TERMINATED
C-----UNLESS USER INPUT PATH CODES, FOR ALL OPERATIONS, SATISFY ALL
C-----VALIDATION TESTS IN SUBROUTINE PATH CHECK.
212  WRITE(LP,2020)
2020  FORMAT(52H ENTER ASSESSMENT MODEL LETTER CODES (A-Z/II/RR/SS):)
      IF(.NOT.ENTR(0)) GO TO 610
      IF(.NOT.QUEST(0)) GO TO 620
C-----
C-----USER TYPED ?, PRODUCE SCENARIO TABLE
C-----WITH HEADER AND FOOTNOTES
      CALL SNMSG(29)
      WRITE(LP,6005) ICD
      CALL INIT(ITMP,28,1,1)
      DO 600 I=1,28
      ITMP=ITST(SNCOD,I)
      IF(ITMP.EQ.0) GO TO 600
      WRITE(LP,6010) SCLST(I)
      CALL SNMSG(I)
600  CONTINUE
      WRITE(LP,6020)
      CALL SNMSG(30)
      CALL SNMSG(31)
      WRITE(LP,6020)
      ISW=1
      GO TO 700
60050FORMAT (/9X,35HSCENARIOS APPROPRIATE FOR CHEMICAL ,A3,22H ARE DESC
1RIBED BELOW -//)
6010  FORMAT (/5X,A10/)
6020  FORMAT (1X)
C-----
C-----USER TYPED (CR), DISPLAY CURRENT MODELS
610  WRITE(LP,6030) (BUFF(I),I=1,K1)
      ISW=1
      GO TO 700
6030  FORMAT (9X,43HAPPROPRIATE HAZARD ASSESSMENT MODELS ARE : ,15A3)
C-----
C-----USER TYPED MODEL INPUT
620  CONTINUE
      DO 213 I=1,15
      UPTH(I)=IBLNK
      IF(.NOT.NAME(TAGX)) GO TO 213
      UPTH(I)=TAGX
213  CONTINUE
211  CALL PAGER(1)
      WRITE(LP,1110) (UPTH(I),I=1,15)
      CALL PTHCK(UPTH,PTLST,MODEL,ISW)
      IF(ISW.EQ.0) GO TO 700
C-----
C-----ERROR RETURNS FOR INVALID PATH CODES
      CALL PAGER(1)
      GO TO(220,230,240,250),ISW
C-----
220  WRITE(LP,1120)
      GO TO 700
230  WRITE(LP,1130)
      GO TO 700
240  WRITE(LP,1140)
      GO TO 700
250  WRITE(LP,1150)
C-----
C-----SECTION TO PRODUCE MODEL TEXT DESCRIPTIONS
700  CALL PAGER(2)
      WRITE(LP,7000)
      IF(.NOT.YESNO(0)) GO TO 750
      CALL PAGER(2)
      WRITE(LP,7010)
710  IF(.NOT.ENTR(0)) GO TO 730
      IF(QUEST(0)) GO TO 730
      IF(.NOT.NAME(TAGX)) GO TO 730

```

```

        IF(TAGX.EQ.EXIT) GO TO 750
        DO 720 I=1,29
        IF(TAGX.NE.PTLST(I)) GO TO 720
        CALL MODEXP(I)
        GO TO 740
720    CONTINUE
        CALL PAGER(2)
        WRITE(LP,7020) TAGX
730    CALL PAGER(2)
        WRITE(LP,7030)
740    CALL PAGER(3)
        WRITE(LP,7040)
        GO TO 710
750    IF(ISW.EQ.0) GO TO 430
        GO TO 212
7000   FORMAT (/40H DO YOU NEED DESCRIPTIONS OF THE MODELS?)
7010   FORMAT (/20H ENTER MODEL LETTER:;)
7020   FORMAT (/1X,A10,11H IS INVALID)
7030   FORMAT (/45H VALID LETTER CODES ARE A TO Z, II, RR, OR SS)
7040   FORMAT (/21H TYPE MODEL LETTER OR/27H TYPE EXIT TO CANCEL REPORT)
C-----READ CHEMICAL RECOGNITION CODE.  TERMINATE IF INSTEAD GET END
C      OF FILE, OTHERWISE AUDIT INPUT THEN STORE.  VALIDATION OCCURS
C      WHEN PROPERTY FILE IS ACCESSED.
500    CONTINUE
510    WRITE(LP,520)
520    FORMAT (36H ENTER OUTPUT UNITS SELECTION (0-4):)
        IF(.NOT.ENTR(0)) GO TO 530
        IF(.NOT.QUEST(0)) GO TO 540
C-----USER TYPED ?
        CALL PAGER(8)
        WRITE(LP,5000)
        GO TO 510
50000   FORMAT (/5X,27H THE AVAILABLE OPTIONS ARE -/5X,16H0 FOR ALL UNITS,/
1 5X,16H1 FOR CGS UNITS,/5X,15H2 FOR SI UNITS,/5X,24H3 FOR ENGLISH
2UNITS, AND/5X,17H4 FOR MIXED UNITS/)
C-----USER TYPED (CR)
530    CALL PAGER(3)
        WRITE(LP,5010) ICVSL
        GO TO 510
5010   FORMAT (/5X,21H CURRENT SELECTION IS ,I1/)
C-----USER TYPED VALUE
540    IF(.NOT.INTEGR(ICVSL)) GO TO 510
        NOFF=0
        IPRAC=1
        IPRRP=0
260    WRITE(LP,2030)
2030   FORMAT(33H ENTER CHEMICAL RECOGNITION CODE:;)
        IF(.NOT.ENTR(0)) GO TO 262
        IF(.NOT.QUEST(0)) GO TO 264
C-----USER TYPED ?
        CALL PAGER(7)
        WRITE(LP,2060)
        GO TO 260
20600   FORMAT (/5X,47H THE CHEMICAL RECOGNITION CODE IS A THREE-LETTER/5X,
1 47HALPHABETIC CODE USED TO SELECT THE CHEMICAL FOR/5X,48H HAZARD A
2ASSESSMENT. REFER TO CHRIS MANUAL II FOR/5X,50H CROSS-REFERENCE LIS
3TS OF CHEMICAL NAMES, SYNONYMS,/5X,22H AND RECOGNITION CODES./)
C-----USER TYPED (CR)
262    CALL PAGER(3)
        WRITE(LP,2070) ICD
        GO TO 260
2070   FORMAT (/5X,15H CURRENT CODE = ,A3/)
C-----USER TYPED CODE
264    IF(.NOT.NAME(ICD)) GO TO 260
        IFLD=1001
261    CALL PAGER(1)
C-----ERROR IF FIELD NUMBER IS NOT 1001, OR RECOGNITION CODE IS BLANK
        IF(IFLD.EQ.1001) GO TO 270
        CALL PAGER(1)
        WRITE(LP,1180)

```

```

      GO TO 30
270  IF(ICD.NE.IBLNK) GO TO 280
      CALL PAGER(1)
      WRITE(LP,1190)
      GO TO 30
C-----STORE RECOGNITION CODE IN STATE FILE
280  IF(NOP.NE.4) GO TO 310
      LIST(1,1)=1001
      LIST(1,2)=0461
      DO 290 I=1,3
        J=I+2
290  LIST(1,J)=IN(I)
      LIST(1,6)=1
      NI=1
      IVAL(1,3)=0
      IVAL(1,2)=0
300  IVAL(1,1)=ICD
      GO TO 320
310  IX=LIST(1,2)/10
      LIST(1,2)=10*IX+1
      IF(NOP.NE.5) LIST(1,2)=LIST(1,2)+4
      GO TO 300
C-----READ BASIC FIELD DATA CARDS, THEN BRANCH ON NOP
320  CONTINUE
      GO TO(330,340,350,355,360),NOP
C-----RUN
330  STCON=3
      GO TO 370
C-----RE-RUN
340  STCON=3
      GO TO 390
C-----CONTINUE
350  IF(STCON.LT.3) STCON=3
      GO TO 390
C-----LOAD DEFAULT
355  CALL PAGER(3)
      WRITE(LP,1220) NF,MNF,NI,MNI
C-----UPDATE DEFAULT
360  IF(CNT.EQ.0) GO TO 410
      IF(IERR.EQ.1) GO TO 410
      STCON=2
      CALL ACCESS(STATE,IDFLT,1,STCON)
      CALL PAGER(5)
      WRITE(LP,1200)
370  DO 380 I=1,2489
380  SAVE(I)=STATE(I)
      SUCON=STCON
C-----AUDIT RUN TIME OPTIONS SELECTED
390  CONTINUE
      CALL LSTFL(2)
      IF(NOP.GT.3) GO TO 400
      INDXX=1
      IF(IPRAC.NE.0) CALL PROP(INDXX)
      IF(INDXX.EQ.1) GO TO 212
      GO TO 260
400  IF(EOFF.EQ.0) GO TO 10
      GO TO 420
410  CALL PAGER(3)
      WRITE(LP,1210)
420  NOP=0
      GO TO 430
C
1000 FORMAT (4A4,3X,3(I1,1X,I1,7X))
1010 FORMAT (//5X,30H*****UNRECOGNIZED CONTROL CARD/5X,4A4)
1020 FORMAT (//5X,19H*****RUN TERMINATED)
1025 FORMAT (10A8)
1030 FORMAT (//5X,39H*****UNEXPECTED END OF FILE ENCOUNTERED)
10400FORMAT (//5X,33HACS STATE FILE INITIALIZED WITH ,A8,29H VALUES, F
      FILE LABEL FOLLOWS -/10X,10A8/)
1050 FORMAT (//5X,37HLISTING OF USER INPUT CARDS FOLLOWS -/5X,37(1H-)/)

```


SUBROUTINE BASIC READS, VALIDATES AND STORES INPUT FIELD DATA CARDS IN THE HACS STATE FILE UNDER CONTROL OF THE ARGUMENT NOP WHICH HAS VALUES ON INPUT AS FOLLOWS -

- 1 RUN OPTION, FIELD CARDS GIVE USER VALUES
- 2 RE-RUN OPTION, FIELD CARDS GIVE USER VALUES
- 3 CONTINUE OPTION, FIELD CARDS GIVE USER VALUES
- 4 LOAD DEFAULT OPTION, FIELD CARDS GIVE DEFAULT VALUES
- 5 UPDATE DEFAULT, FIELD CARDS GIVE DEFAULT VALUES

THE ROUTINE PROCESSES ALL FIELD DATA CARDS (EXCEPT FOR FIELD 1001 GIVING THE CHEMICAL RECOGNITION CODE) UNTIL EITHER A FIELD 9999 CARD OR AN END OF FILE IS ENCOUNTERED. ON RETURN, CNT GIVES THE COUNT OF FIELD DATA CARDS READ, AND IERR IS 0 IF NO ERRORS WERE IDENTIFIED. IERR IS SET TO ONE IF AT LEAST ONE ERROR WAS DETECTED. ON RETURN, THE INDICATOR EOF IS SET TO -1 IF AN END OF FILE WAS ENCOUNTERED, 0 OTHERWISE. AN OUTPUT LISTING OF EACH DATA CARD IS PRODUCED, FOLLOWED BY ERROR MESSAGES AND/OR FIELD VALUE CONVERSIONS, IF ANY. THE GENERAL NATURE OF THE PROCESSING FOR EACH TYPE OF FIELD DATA CARD IS INDICATED BELOW -

FOR NOP = 1, 2, OR 3

READ USER DATA CARD CONTAINING

IFLD,FVL,TAG ... WHERE -

1. IFLD IS EITHER 9999 TO DENOTE END OF DATA, OR IFLD IS A PREVIOUSLY DEFINED FIELD NUMBER IN THE RANGE 2 TO 9998.
2. TAG IS EITHER BLANK OR MATCHES AN ALLOWABLE UNIT LABEL FOR THE TYPE OF PHYSICAL QUANTITY DEFINED FOR FIELD IFLD. IF TAG IS BLANK, THE VALUE FVL IS ASSUMED TO BE IN INTERNAL (CGS) UNITS AND IS NOT CONVERTED. FIELD VALUE CONVERSIONS ARE APPLIED ONLY TO REAL FIELD VALUES, NOT INTEGERS
3. IF THE VALUE FVL EXCEEDS THE NOMINAL BOUNDS DEFINED IN THE DEFAULT FILE, A WARNING MESSAGE IS GENERATED BUT THE VALUE IS STORED. HOWEVER, FOR INTEGER FIELDS, IF THE VALUE GIVEN EXCEEDS AN INTERNAL LIMIT (ITOL), A FIELD JUSTIFICATION ERROR IS ASSUMED. IN THIS CASE AN ERROR MESSAGE IS GENERATED AND THE FIELD VALUE IS NOT STORED.

FOR NOP = 4, READ DEFAULT DATA CARD CONTAINING

IFLD,FVL,TAG,IVAR,MM,FMN,FMX,ITYP,(IN(I),I=1,3)...WHERE -

4. IFLD IS EITHER 9999 TO DENOTE END OF DATA, OR IFLD IS A VALID FIELD NUMBER IN THE RANGE 2 TO 9998 WHICH HAS NOT BEEN DEFINED BY A PREVIOUSLY ENTERED CARD.
5. THE INDICATOR IVAR DEFINES THE INTERNAL STORAGE MODE FOR THE FIELD VALUE (0 FOR INTEGER, 1 FOR REAL)
6. THE MIN/MAX SELECTOR MM MUST BE GIVEN AS 3 INDICATING THAT BOTH MINIMUM AND MAXIMUM NOMINAL ROUNDS ARE GIVEN FOR THE DEFAULT FILE DEFINITION.
7. THE FIELD NAME, 1 TO 12 ALPHANUMERIC CHARACTERS READ INTO (IN(I),I=1,3), MUST NOT BE BLANK.
8. THE TYPE OF PHYSICAL QUANTITY GIVEN BY ITYP MUST BE IN THE RANGE 1 TO MTP, CORRESPONDING TO A PRE-DEFINED QUANTITY TYPE.
9. SEE NOTE 2 ABOVE FOR VALIDATION OF THE UNIT LABEL TAG. ALSO, FOR DEFAULT DATA, INPUT CONVERSIONS ARE APPLIED TO THE FIELD VALUE AND BOTH RANGE LIMITS FOR REAL FIELDS. INPUT CONVERSIONS ARE NOT APPLIED TO INTEGER FIELDS.
10. IF THE FIELD VALUE FVL IS NOT WITHIN THE BOUNDS GIVEN, AN ERROR MESSAGE IS GENERATED, AND THE FIELD DEFINITION IS CANCELLED. ALSO, FOR INTEGER FIELDS, THE VALUE AND RANGE LIMITS ARE COMPARED TO A LIMITING VALUE ITOL TO TEST FOR INCORRECT FIELD JUSTIFICATION.

FOR NOP = 5, READ DEFAULT FILE UPDATE CARD CONTAINING

IFLD,FVL,TAG,MM,FMN,FMX,(IN(I),I=1,3) ... WHERE -

11. SEE NOTE 1 ABOVE.

12. THE MIN/MAX SELECTOR MM IS GIVEN AS

- 0 NEITHER LIMIT UPDATED
- 1 MINIMUM VALUE GIVEN
- 2 MAXIMUM VALUE GIVEN
- 3 NEW VALUES GIVEN FOR BOTH LIMITS

13. THE FIELD NAME (IN(I),I=1,3) IS REPLACED BY THE USER SPECIFIED LABEL IF NON-BLANK. OTHERWISE THE NAME STORED IN THE DEFAULT FILE IS UNCHANGED.

14. SEE NOTE 2 ABOVE. IN ADDITION, INPUT CONVERSIONS, IF ANY, ARE APPLIED TO ALL RANGE LIMITS GIVEN FOR REAL FIELDS (AS SPECIFIED BY MM)

15. SEE NOTE 10 ABOVE. RANGE LIMIT TESTS USE UPDATED VALUES AS REQUESTED.

CNT = ARGUMENT, INTEGER COUNT OF FIELD DATA CARDS READ EXCLUDING 9999 OR END OF FILE TERMINATORS

CR = INTEGER FORTRAN UNIT NUMBER FOR CARD READER

D = CONVERSION FACTOR FOR REAL FIELD OBTAINED FROM DATA ARRAY CONV

EOF = ARGUMENT, RETURNED AS -1 IF AN END OF FILE WAS ENCOUNTERED WHEN READING FIELD DATA CARDS. OTH WISE RETURNED AS 0.

FAC = ARRAY OF CONVERSION FACTORS FOR TEMPERATURE FIELDS

FBLNK = REAL DATA WORD CONTAINING ALL BLANKS USED TO TEST UNIT LABEL FIELD TAG

FMN = LOWER LIMIT RANGE VALUE ENTERED ON USER INPUT CARD IN FIXED POINT FORMAT

FMX = UPPER LIMIT RANGE VALUE ENTERED ON USER INPUT CARD IN FIXED POINT FORMAT

FVL = FIELD VALUE ENTERED ON USER INPUT CARD IN FIXED POINT FORMAT

I = DUMMY SUBSCRIPT

IBLNK = INTEGER DATA WORD CONTAINING ALL BLANKS USED TO TEST FIELD NAME INPUT

IERR = ARGUMENT, RETURNED AS 1 IF ONE OR MORE INPUT ERRORS WERE FOUND, 0 OTHERWISE.

IFLD = FIELD NUMBER ENTERED ON USER INPUT CARD

ILN = INDEX TO POSITION IN STATE FILE FIELD DEFINITION ARRAY CORRESPONDING TO FIELD IFLD.

IMN = LOWER LIMIT RANGE VALUE FOR INTEGER FIELD, OBTAINED FROM DEFAULT FILE OR CONVERSION OF INPUT VALUE FMN

IMX = UPPER LIMIT RANGE VALUE FOR INTEGER FIELD, OBTAINED FROM DEFAULT FILE OR CONVERSION OF INPUT VALUE FMX

IN = ARRAY FOR STORAGE OF USER SPECIFIED FIELD NAME CONTAINING UP TO 12 CHARACTERS STORED AS 3A4.

ISRC = FIELD VALUE SOURCE CODE IN STATE FILE BEFORE USER FIELD TRANSACTION IS STORED

ISW = CONTROL SWITCH SET TO 0 IF INPUT FOR FIELD NAME IS BLANK, 1 IF NON-BLANK, FOR HANDLING OPTIONAL CHANGE ON DEFAULT FILE UPDATE

ISYS = INTEGER IN RANGE 1 TO MSYS DETERMINED BY MATCHING UNIT TAG TO PRE-DEFINED LABELS FOR THE PARTICULAR TYPE OF PHYSICAL QUANTITY. IF TAG IS BLANK OR THE SAME AS INTERNAL MACS UNITS, ISYS IS COMPUTED AS 1 AND CONVERSION ON INPUT IS SKIPPED.

ITOL = INTEGER TOLERANCE LEVEL SET AS DATA VALUE FOR TESTING FIELD JUSTIFICATION OF INTEGER VALUES ON INPUT

ITYP = TYPE OF PHYSICAL QUANTITY FOR INDIVIDUAL FIELD IN RANGE 1 TO MTYP

IVAR = INTERNAL FIELD VALUE STORAGE MODE INDICATOR (0 FOR INTEGER, 1 FOR REAL)

IVL = FIELD VALUE FOR INTEGER FIELD, OBTAINED FROM INPUT CARD BY CONVERSION OF FVL

IX = INDEX TO POSITION IN FIELD VALUE DATA ARRAYS FVAL OR IVAL CORRESPONDING TO FIELD DEFINITION IN ARRAY LIST

J = DUMMY SUBSCRIPT

JSYS = INDEX TO CONVERSION FACTOR DATA ARRAY CORRESPONDING TO ISYS-1 WHERE ISYS INDEXES THE UNIT LABEL DATA

MM = MIN/MAX SELECTOR USED TO SPECIFY WHICH RANGE VALUES

C HAVE BEEN ENTERED ON USER DATA CARD (0=NEITHER,
 C 1=MIN ONLY, 2=MAX ONLY, 3=BOTH MIN AND MAX)
 C NFLD = TOTAL NUMBER OF FIELDS DEFINED IN HACS STATE FILE
 C NOP = CONTROL SELECTOR DETERMINED FROM USER SPECIFIED
 C PROCESSING OPTION
 C SEQ = REAL VARIABLE USED TO READ AND DISPLAY CONTENTS OF
 C SEQUENCE FIELD ON INPUT CARD
 C TAG = FIELD ON USER INPUT CARD OR UNIT LABEL IN A8 FORMAT
 C
 C COMMON VARIABLES USED - CONV,FVAL,IVAL,LIST,LP,MNF,MNI,MSYS,
 C MTYP,NF,NI,UNIT
 C
 C SUBROUTINES REQUIRED - IABS,IFEOF,IFIX,PAGER
 C
 C AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
 C 35/309A ACORN PARK
 C CAMBRIDGE, MASS., 02140
 C TEL. 617-864-5770 EXT. 2813
 C
 C DATE - 12 FEBRUARY 1976

C
 C OCOMMON/BASE/SAVE(2489),UPTH(15),MSG(10),MNF,MNI,
 C 1 NF,NI,LIST(275,6),FVAL(225,3),IVAL(50,3)
 C INTEGER UPTH
 C REAL MSG
 C DIMENSION STATE(2489)
 C EQUIVALENCE (STATE(1),MSG(1))
 C
 C COMMON/CNVDT/CONV(3,47),MSYS,MTYP,UNIT(4,47)
 C
 C COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
 C
 C
 C INTEGER CNT,CR,EOFF
 C DIMENSION FAC(3),IN(3)
 C DATA CR/60/,FBLNK/8H /,IBLNK/4H /,ITOL/10000/
 C DATA (FAC(I),I=1,3)/1.0,1.8,1.0/
 C
 C C-----INITIALIZATION SECTION. SET VARIABLE VALUES, THEN BRANCH ON
 C OPERATION FLAG TO PRINT HEADER FOR DATA INPUT LISTING
 C
 C CNT=0
 C EOFF=0
 C IERR=0
 C NFLD=NF+NI
 C CALL PAGER(3)
 C GO TO(10,10,10,20,30),NOP
 C 10 WRITE(LP,1000)
 C GO TO 40
 C 20 WRITE(LP,1010)
 C GO TO 40
 C 30 WRITE(LP,1020)
 C
 C C-----RETURN HERE TO READ EACH NEW USER INPUT CARD
 C 40 READ(CR,1030) IFLD,FVL,TAG,IVAR,MH,FMN,FHX,ITYP,(IN(I),I=1,3),SEQ
 C
 C C-----TEST FOR TERMINATION. RETURN ON END OF FILE OR END OF DATA
 C CARD. OTHERWISE INCREMENT COUNT OF USER DATA CARDS READ.
 C IF(EOF(CR))50,60
 C 50 EOFF=-1
 C CALL PAGER(3)
 C WRITE(LP,1040)
 C RETURN
 C 60 IF(IFLD.NE.9999) GO TO 70
 C CALL PAGER(3)
 C WRITE(LP,1050)
 C RETURN
 C 70 CNT=CNT+1
 C
 C C-----LOOK UP FIELD NUMBER READ ON INPUT CARD IN HACS STATE FILE
 C ILN=1
 C 80 ILN=ILN+1

```

      IF(ILN.GT.NFLD) GO TO 85
      IF(IFLD.EQ.LIST(ILN,1)) GO TO 130
      GO TO 80
    85 CONTINUE
C-----FIELD NUMBER WAS NOT FOUND IN HACS STATE FILE. INDEX ILN
C      POINTS TO NEXT AVAILABLE SLOT IN FILE FOR ENTERING DEFAULT
C      DATA. AN ERROR EXISTS FOR OTHER OPERATIONS.
      IF(NOP.EQ.4) GO TO 110
      CALL PAGER(1)
      WRITE(LP,1060) IFLD
    90 CALL PAGER(1)
      WRITE(LP,1070)
      IERR=1
      GO TO 40
C-----DISPLAY CONTENTS OF DATA CARD READ TO CREATE DEFAULT FILE
C      ENTRY. USE EITHER INTEGER OR REAL (BY DEFAULT) FORMAT
    110 CALL PAGER(1)
      IF(IVAR.EQ.0) GO TO 120
      WRITE(LP,1080) IFLD,FVL,TAG,IVAR,MM,FMN,FMX,ITYP,(IN(I),I=1,3),SEQ
      GO TO 180
    120 IVL=IFIX(FVL)
      IMN=IFIX(FMN)
      IMX=IFIX(FMX)
      WRITE(LP,1090) IFLD,IVL,TAG,IVAR,MM,IMN,IMX,ITYP,(IN(I),I=1,3),SEQ
      GO TO 180
C-----FIELD NUMBER READ FROM INPUT MATCHED ENTRY IN STATE FILE.
C      ERROR CONDITION FOR ENTERING DEFAULT DATA, OTHERWISE UNCODE
C      INFORMATION IN STATE FILE AND AUDIT INPUT.
    130 IF(NOP.NE.4) GO TO 140
      CALL PAGER(1)
      WRITE(LP,1100) IFLD
      GO TO 90
    140 IVAR=LIST(ILN,2)/1000
      ISRC=1000*IVAR
      ITYP=(LIST(ILN,2)-ISRC)/10
      ISRC=LIST(ILN,2)-10*ITYP-ISRC
C
      IX=LIST(ILN,6)
      CALL PAGER(1)
      IF(NOP.NE.5) GO TO 160
      IF(IVAR.EQ.0) GO TO 150
      WRITE(LP,1110) IFLD,FVL,TAG,MM,FMN,FMX,(IN(I),I=1,3),SEQ
      GO TO 210
    150 IVL=IFIX(FVL)
      IMN=IFIX(FMN)
      IMX=IFIX(FMX)
      WRITE(LP,1120) IFLD,IVL,TAG,MM,IMN,IMX,(IN(I),I=1,3),SEQ
      GO TO 210
C
    160 IF(IVAR.EQ.0) GO TO 170
      FMN=FVAL(IX,2)
      FMX=FVAL(IX,3)
      WRITE(LP,1080) IFLD,FVL,TAG
      GO TO 280
    170 IVL=IFIX(FVL)
      IMN=IVAL(IX,2)
      IMX=IVAL(IX,3)
      WRITE(LP,1090) IFLD,IVL,TAG
      GO TO 280
C-----TEST FOR VALID FIELD NUMBER
C
    180 IF(IFLD.LT.1) GO TO 190
      IF(IFLD.LE.9998) GO TO 200
    190 CALL PAGER(1)
      WRITE(LP,1130)
      GO TO 90
C-----TEST FOR VALID STORAGE MODE INDICATOR. NOTE THAT IF IVAR=0,
C      IVL, IMN AND IMX HAVE ALREADY BEEN OBTAINED FOR AUDIT.

```

```

200 IF(IVAR.EQ.0) GO TO 210
    IF(IVAR.EQ.1) GO TO 210
    CALL PAGER(1)
    WRITE(LP,1140)
    GO TO 90
C
C-----TEST MINIMUM/MAXIMUM INPUT DATA SELECTOR
210 IF(MM.LT.0) GO TO 220
    IF(MM.GT.3) GO TO 220
    IF(NOP.EQ.5) GO TO 230
    IF(MM.EQ.3) GO TO 230
220 CALL PAGER(1)
    WRITE(LP,1150)
    GO TO 90
C
C-----TEST FOR NON-BLANK FIELD NAME (ISW=1).
230 ISW=1
    DO 240 I=1,3
    IF(IN(I).NE.IBLNK) GO TO 250
240 CONTINUE
    ISW=0
    IF(NOP.EQ.5) GO TO 290
    CALL PAGER(1)
    WRITE(LP,1160)
    GO TO 90
250 IF(NOP.EQ.5) GO TO 290
C
C-----TEST TYPE CODE. MUST BE IN RANGE OF DEFINED TYPES OF
C      PHYSICAL QUANTITIES.
    IF(ITYP.LT.1) GO TO 270
    IF(ITYP.LE.MTYP) GO TO 290
270 CALL PAGER(1)
    WRITE(LP,1170) MTYP
    GO TO 90
C
C-----SET PARAMETERS FOR NOP = 1, 2 OR 3
280 MM=0
    ISW=0
C
C-----VERIFY UNIT LABEL SPECIFICATION. CONVERSION TO INTERNAL UNITS
C      IS REQUIRED UNLESS ISYS IS DETERMINED TO BE 1.
290 ISYS=1
    IF(TAG.EQ.FBLNK) GO TO 310
300 IF(TAG.EQ.UNIT(ISYS,ITYP)) GO TO 310
    ISYS=ISYS+1
    IF(ISYS.LE.MSYS) GO TO 300
    CALL PAGER(1)
    WRITE(LP,1180)
    GO TO 90
C
C-----SEPARATE PROCESSING FOR INTEGER AND REAL FIELDS
310 IF(IVAR.EQ.0) GO TO 450
C
C-----CONVERT REAL FIELD VALUES IF NECESSARY
    IF(ISYS.EQ.1) GO TO 330
    JSYS=ISYS-1
    D=CONV(JSYS,ITYP)
    IF(ITYP.NE.6) GO TO 315
    FVL=(FVL-D)/FAC(JSYS)
    GO TO 316
315 FVL=D*FVL
316 CALL PAGER(1)
    IF(NOP.GT.3) GO TO 320
    WRITE(LP,1190) FVL,UNIT(1,ITYP)
    GO TO 330
320 IF(ITYP.NE.6) GO TO 321
    FMN=(FMN-D)/FAC(JSYS)
    FMX=(FMX-D)/FAC(JSYS)
    GO TO 322
321 FMN=D*FMN

```

```

      FMX=D*FMX
322 WRITE(LP,1190) FVL,UNIT(1,ITYP),FMN,FMX
C-----FOR OPERATIONS TO CHANGE DEFAULT VALUES, RESTORE UNCHANGED
C      LIMITS FROM DEFAULT FILE
330 IF(NOP.NE.5) GO TO 360
      IF(MH.NE.0) GO TO 350
      FMN=FVAL(IX,2)
340 FMX=FVAL(IX,3)
      GO TO 360
350 IF(MH.EQ.3) GO TO 360
      IF(MH.EQ.1) GO TO 340
      FMN=FVAL(IX,2)
C-----CHECK RANGE OF REAL VARIABLE
360 IF(FVL.LT.FMN) GO TO 370
      IF(FVL.LE.FMX) GO TO 380
370 CALL PAGER(1)
      WRITE(LP,1200) FMN,FMX,UNIT(1,ITYP)
      IF(NOP.GT.3) GO TO 90
C-----STORE REAL VALUES IN STATE FILE
380 IF(NOP.EQ.4) GO TO 390
      IF(NOP.EQ.5) GO TO 410
      FVAL(IX,1)=FVL
      LIST(ILN,2)=LIST(ILN,2)-ISRC+5
      GO TO 40
390 IF(NF.LT.MNF) GO TO 400
      CALL PAGER(1)
      WRITE(LP,1210) MNF
      GO TO 90
400 NF=NF+1
      NFLD=NFLD+1
      IX=NF
      LIST(ILN,1)=IFLD
      LIST(ILN,2)=1001+10*ITYP
      LIST(ILN,6)=NF
      GO TO 420
410 LIST(ILN,2)=LIST(ILN,2)-ISRC+1
420 FVAL(IX,1)=FVL
      FVAL(IX,2)=FMN
      FVAL(IX,3)=FMX
430 IF(ISW.EQ.0) GO TO 40
      DO 440 I=1,3
      J=I+2
440 LIST(ILN,J)=IN(I)
      GO TO 40
C-----TEST FOR REQUESTED CONVERSION OF INTEGER FIELD VALUES
450 IF(ISYS.EQ.1) GO TO 460
      CALL PAGER(1)
      WRITE(LP,1220)
C-----TEST FOR CORRECT FIELD JUSTIFICATION OF INTEGER VALUES
460 IF(IABS(IVL).GT.ITOL) GO TO 470
      IF(NOP.LE.3) GO TO 480
      IF(IABS(IMN).GT.ITOL) GO TO 470
      IF(IABS(IMX).LE.ITOL) GO TO 480
470 CALL PAGER(1)
      WRITE(LP,1230) ITOL
      GO TO 90
C-----COMPARE INTEGER FIELD VALUE TO NOMINAL RANGE LIMITS. FOR
C      OPERATIONS TO CHANGE DEFAULT VALUES, FIRST RESTORE UNCHANGED
C      LIMITS FROM DEFAULT FILE.
480 IF(NOP.NE.5) GO TO 510
      IF(MH.NE.0) GO TO 500
      IMN=IVAL(IX,2)
490 IMX=IVAL(IX,3)
      GO TO 510
500 IF(MH.EQ.3) GO TO 510

```

```

      IF(MM.EQ.1) GO TO 490
      IMN=IVAL(IX,2)
C-----COMPARE TO NOMINAL LIMITS
510 IF(IVL.LT.IMN) GO TO 520
    IF(IVL.LE.IMX) GO TO 530
520 CALL PAGER(1)
    WRITE(LP,1240) IMN,IMX,UNIT(1,ITYP)
    IF(NOP.GT.3) GO TO 90
C-----STORE INTEGER VALUES IN STATE FILE
530 IF(NOP.EQ.4) GO TO 540
    IF(NOP.EQ.5) GO TO 560
    IVAL(IX,1)=IVL
    LIST(ILN,2)=LIST(ILN,2)-ISRC+5
    GO TO 40
540 IF(NI.LT.MNI) GO TO 550
    CALL PAGER(1)
    WRITE(LP,1250) MNI
    GO TO 90
550 NI=NI+1
    NFLD=NFLD+1
    IX=NI
    LIST(ILN,1)=IFLD
    LIST(ILN,2)=1+10*ITYP
    LIST(ILN,6)=NI
    GO TO 570
560 LIST(ILN,2)=LIST(ILN,2)-ISRC+1
570 IVAL(IX,1)=IVL
    IVAL(IX,2)=IMN
    IVAL(IX,3)=IMX
    GO TO 430
C
10000FORMAT (5X,5HFIELD/5X,31HNUMBER FIELD VALUE UNIT/5X,
1 6(1H-),2X,15(1H-),2X,8(1H-))
10100FORMAT (5X,5HFIELD,30X,8H1=0 MIN/5X,95HNUMBER FIELD VALUE
1 UNIT R=1 MAX NOMINAL MINIMUM NOMINAL MAXIMUM TYPE FIELD N
2AME,4X,7HCOMMENT/5X,6(1H-),2X,15(1H-),2X,8(1H-),2(2X,3(1H-)),
3 2(2X,15(1H-)),2X,4(1H-),2X,12(1H-),2X,7(1H-))
10200FORMAT (5X,5HFIELD,30X,8H MIN/5X,95HNUMBER FIELD VALUE
1 UNIT MAX NOMINAL MINIMUM NOMINAL MAXIMUM FIELD N
2AME,4X,7HCOMMENT/5X,6(1H-),2X,15(1H-),2X,8(1H-),2(2X,3(1H-)),
3 2(2X,15(1H-)),2X,4(1H-),2X,12(1H-),2X,7(1H-))
1030 FORMAT (I4,F15.0,A8,2I1,2F15.0,I2,3A4,A7)
1040 FORMAT (5X,11HEND OF FILE//)
1050 FORMAT (6X,4H9999//)
1060 FORMAT (5X,39H*****ERROR - UNDEFINED FIELD NUMBER = (,I4,1H))
1070 FORMAT (10X,21HINPUT DATA IS IGNORED)
10800FORMAT (6X,I4,5X,G13.4,2X,A8,3X,I1,4X,I1,5X,G13.4,4X,G13.4,3X,
1 I2,3X,3A4,2X,A7)
10900FORMAT (6X,I4,8X,I10,2X,A8,3X,I1,4X,I1,8X,I10,7X,I10,3X,I2,3X,
1 3A4,2X,A7)
11000FORMAT (5X,48H*****ERROR - PREVIOUSLY DEFINED FIELD NUMBER = (,
1 I4,1H))
1110 FORMAT (6X,I4,5X,G13.4,2X,A8,8X,I1,5X,G13.4,4X,G13.4,8X,3A4,2X,A7)
1120 FORMAT (6X,I4,8X,I10,2X,A8,8X,I1,8X,I10,7X,I10,8X,3A4,2X,A7)
11300FORMAT (5X,65H*****ERROR - ILLEGAL FIELD NUMBER NOT IN RANGE 1 TO
1 19998, OR 9999)
11400FORMAT (5X,73H*****ERROR - STORAGE MODE INDICATOR NOT 0 FOR INTEGE
1R OR 1 FOR REAL FIELD)
1150 FORMAT (5X,44H*****ERROR - ILLEGAL MIN/MAX INDICATOR VALUE)
1160 FORMAT (5X,34H*****ERROR - FIELD NAME IS MISSING)
11700FORMAT (5X,57H*****ERROR - TYPE OF PHYSICAL QUANTITY NOT IN RANGE
1 11 TO ,I2)
1180 FORMAT (5X,51H*****ERROR - UNIT LABEL INCONSISTENT WITH TYPE CODE)
1190 FORMAT (11X,1H=,4X,G13.4,2X,A8,14X,G13.4,4X,G13.4)
12000FORMAT (5X,46H*****WARNING/ERROR - FIELD VALUE NOT IN RANGE ,
1 G13.4,4H TO ,G13.4,1X,A8)
12100FORMAT (5X,60H*****ERROR - NUMBER OF REAL FIELDS DEFINED EXCEEDS L
1IMIT OF ,I3)
12200FORMAT (5X,64H*****WARNING - CONVERSION OF INTEGER FIELD VALUES NO
1T APPLICABLE)

```

```

12300FORMAT (5X,70H*****ERROR - INTEGERS MUST BE RIGHT JUSTIFIED, VALUE
1 EXCEEDS LIMIT OF ,I10)
12400FORMAT (5X,46H*****WARNING/ERROR - FIELD VALUE NOT IN RANGE ,
1I10,4H TO ,I10,1X,A8)
12500FORMAT (5X,63H*****ERROR - NUMBER OF INTEGER FIELDS DEFINED EXCEED
1S LIMIT OF ,I3)

```

C

```

END
SUBROUTINE BNDCK(ISW,TEMP,ITEMP,NFTLO,NFTUP)

```

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

```

SUBROUTINE BOUND CHECK COMPARES THE REQUESTED TEMPERATURE TEMP
TO THE BOUNDS, IF ANY, DEFINED FOR A TEMPERATURE FUNCTION. THE
ARGUMENT ISW PROVIDES A CONTROL PARAMETER SO THAT THE
TEMPERATURE BOUNDS, DEFINED AS FIELDS NFTLO AND NFTUP IN THE
STATE FILE, ARE RETRIEVED ONLY ONCE AND STORED FOR LATER USE
IN COMMON AS THE VALUES TLO AND TUP, HAVING SOURCE CODES ILO
AND IUP, RESPECTIVELY. A MESSAGE IS GENERATED IF THE REQUESTED
TEMPERATURE EXCEEDS A BOUNDING VALUE. THE TEMPERATURE VALUE
IS SET TO THE LIMIT ONLY IF THE EXCEEDED BOUND IS GIVEN WITH
A SOURCE CODE OF GREATER THAN DEFAULT. IF ADJUSTED THE
TEMPERATURE SOURCE CODE IS CHANGED TO NOT EXCEED AN ESTIMATED
PROPERTY VALUE. THE SOURCE CODE FOR THE COMPUTED FUNCTION
VALUE IS OBTAINED AS THE MINIMUM OF ALL SOURCE CODES ON THE
RIGHT-HAND SIDE OF THE TEMPERATURE FUNCTION EQUATION. THUS
IF ANY DEFAULT VALUES HAVE BEEN USED, THE TEMPERATURE FUNCTION
VALUE WILL BE COMPUTED FOR DISPLAY, BUT THE PRIORITY STRUCTURE
WILL NOT PERMIT THE COMPUTED VALUE TO BE SAVED IN THE STATE
FILE. FINALLY, THE ROUTINE WRITES THE VALUE OF TEMPERATURE
TO BE USED IN THE COMPUTATION.

```

```

IERR = DATA BASE RECALL ERROR INDICATOR, NOT USED. ERRORS.
      IF ANY, ALSO RETURN SOURCE CODE OF ZERO WHICH IS
      USED INSTEAD OF SEPARATE ERROR PROCESSING
ISW = ARGUMENT, SET TO 1 ON FIRST CALL TO FORCE RETRIEVAL
      OF TEMPERATURE BOUNDS FROM STATE FILE AND STORAGE
      IN COMMON. SET TO 2 ON SECOND CALL TO USE VALUES
      PREVIOUSLY STORED IN COMMON.
ITEMP = ARGUMENT, SOURCE CODE ASSOCIATED WITH TEMPERATURE TO
      BE USED IN FUNCTION CALCULATION
NFTLO = FIELD NUMBER IN STATE FILE FOR LOWER TEMPERATURE
      LIMIT (USED ONLY IF ISW=1)
NFTUP = FIELD NUMBER IN STATE FILE FOR UPPER TEMPERATURE
      LIMIT (USED ONLY IF ISW=1)
TEMP = ARGUMENT, TEMPERATURE AT WHICH FUNCTION IS TO BE
      COMPUTED

```

```

COMMON VARIABLES USED - ILO,IMN,IT,IUP,LP,NPRRP,T,TLO,TUP

```

```

SUBROUTINES REQUIRED - FRCL,PAGER

```

```

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
        35/309A ACORN PARK,
        CAMBRIDGE, MASS., 02140
        TEL. 617-864-5770 EXT. 2813

```

```

DATE - 1 APRIL 1976

```

```

COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)

```

```

COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP

```

```

COMMON/TFUN/A,B,C,D,ILO,IMN,IT,IUP,T,TLO,TUP

```

```

-----BRANCH ON ARGUMENT CONTROL SWITCH, AND RECALL TEMPERATURE
RANGE FOR FUNCTION ON FIRST TIME THROUGH. NOTE THAT SOURCE
CODES ARE SAVED SEPARATELY AND ARE ZERO IF AN ERROR OCCURRED.

```

```

IF (ISW.EQ.2) GO TO 10
ILO=7
CALL FRCL(NFTLO,TLO,ILO,IERR)
IUP=7
CALL FRCL(NFTUP,TUP,IUP,IERR)

```



```

C      IERR  = DATABASE RECALL ERROR INDICATOR SET TO ONE IF AN
C      ERROR OCCURRED.
C      IMX   = HIGHEST SOURCE CODE FOR COEFFICIENTS REQUESTED
C      ISRC  = DUMMY ARGUMENT RETURNING SOURCE CODE FOR SINGLE
C      FIELD VALUE RECALL USED TO OBTAIN MIN/MAX SOURCE
C      CODES FOR MULTIPLE RECALLS
C      NFA   = FIELD NUMBER FOR COEFFICIENT TO BE STORED AS A
C      NFB   = FIELD NUMBER FOR COEFFICIENT TO BE STORED AS B
C      NFC   = FIELD NUMBER FOR COEFFICIENT TO BE STORED AS C
C      NFD   = FIELD NUMBER FOR COEFFICIENT TO BE STORED AS D

```

COMMON VARIABLES USED - A,B,C,D,IMN,LP,NPRRP

SUBROUTINES REQUIRED - FRCL,PAGER

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
 35/309A ACORN PARK,
 CAMBRIDGE, MASS., 02140
 TEL. 617-864-5770 EXT. 2813

DATE - 1 APRIL 1976

COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)

COMMON/IOCNT/ICVSL, IPRAC, IPRRP, NOFF, NPRRP

COMMON/TFUN/A,B,C,D,ILO,IMN,IT,IUP,T,TLO,TUP

C-----INITIALIZE

```

C      COEF=.FALSE.
C      IERR=0
C      IMN=7
C      IMX=-1

```

C-----RECALL FIRST COEFFICIENT VALUE

```

C      IF(NFA.LE.0) GO TO 20
C      ISRC=7
C      CALL FRCL(NFA,A,ISRC,IERR)
C      IF(ISRC.LT.IMN) IMN=ISRC
C      IF(ISRC.GT.IMX) IMX=ISRC

```

C-----RECALL SECOND COEFFICIENT VALUE

```

C      20 IF(NFB.LE.0) GO TO 30
C      ISRC=7
C      CALL FRCL(NFB,B,ISRC,IERR)
C      IF(ISRC.LT.IMN) IMN=ISRC
C      IF(ISRC.GT.IMX) IMX=ISRC

```

C-----RECALL THIRD COEFFICIENT VALUE

```

C      30 IF(NFC.LE.0) GO TO 40
C      ISRC=7
C      CALL FRCL(NFC,C,ISRC,IERR)
C      IF(ISRC.LT.IMN) IMN=ISRC
C      IF(ISRC.GT.IMX) IMX=ISRC

```

C-----RECALL FOURTH COEFFICIENT VALUE

```

C      40 IF(NFD.LE.0) GO TO 50
C      ISRC=7
C      CALL FRCL(NFD,D,ISRC,IERR)
C      IF(ISRC.LT.IMN) IMN=ISRC
C      IF(ISRC.GT.IMX) IMX=ISRC

```

C-----TEST FOR ERROR CONDITIONS OR USE OF DEFAULT VALUES.

```

C      50 IF(IERR.NE.0) GO TO 60
C      IF(IMX.LE.1) GO TO 70
C      RETURN

```

ERROR IN RECALL OF FIELD VALUES

PAGE 2
 P. 1000

```

C
C-----ALL COEFFICIENTS EITHER MISSING OR DEFAULT VALUES
70 IF(NPRRP.EQ.0) GO TO 90
   CALL PAGER(2)
   WRITE(LP,1010)
C
90 COEF=.TRUE.
   RETURN
C
10000FORMAT (/ 5X,60H****ERROR - COMPUTATION OF FUNCTION VALUE UNABLE
1TO PROCEED/)
1010 FORMAT ( 10X,33HTEMPERATURE FUNCTION IS UNDEFINED/)
   END
   FUNCTION ICOMP(ICODA,ICODB)

C
C      FUNCTION ICOMP PERFORMS A COMPARISON OF TWO CHEMICAL
C      RECOGNITION CODES, ICODA AND ICODB, AND RETURNS AN INTEGER
C      VALUE AS FOLLOWS -
C      ICOMP = -1 ,   ICODA .LT. ICODB
C      ICOMP = 0 ,   ICODA .EQ. ICODB
C      ICOMP = +1 ,   ICODA .GT. ICODB
C
C      THE COMPARISON REFLECTS THE SEQUENCE OF INTERNAL CHARACTER
C      CODES AND, FOR THREE CHARACTER NON-BLANK CODES, WILL PROVIDE
C      THE DESIRED ALPHABETIC SEQUENCE. THIS ROUTINE ASSUMES THAT
C      THE CODES ARE STORED IN A3 FORMAT - THEREFORE BY DEFINITION
C      A BLANK (INTERNAL CODE 60) IS STORED IN THE FOURTH CHARACTER
C      POSITION. SINCE THE LAST BIT IS ALWAYS ZERO, WORD CONTENTS
C      ARE SHIFTED ONE BIT POSITION RIGHT, THEN MANIPULATED TO DROP
C      THE SIGN BIT WHICH IS PRESENT FROM J TO Z. THE FINAL
C      COMPARISON IS MADE ON BIT PATTERNS EQUIVALENT TO THOSE ON
C      INPUT, BUT SHIFTED BY ONE POSITION.
C
C      ICODA = THREE CHARACTER RECOGNITION CODE, ARGUMENT
C      ICODB = THREE CHARACTER RECOGNITION CODE, ARGUMENT
C      ICOMP = INTEGER FUNCTION VALUE
C      IXA   = INTERNAL VARIABLE USED FOR MANIPULATION OF CONTENTS
C              OF ARGUMENT ICODA
C      IXB   = INTERNAL VARIABLE USED FOR MANIPULATION OF CONTENTS
C              OF ARGUMENT ICODB
C
C      COMMON VARIABLES USED - NONE
C      SUBROUTINES REQUIRED - NONE
C
C      AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.
C              35/309A ACORN PARK,
C              CAMBRIDGE, MASS., 02140
C              TEL. 617-864-5770 EXT. 2813
C      DATE - 27 SEPTEMBER 1975
C
C-----COPY RECOGNITION CODES INTO INTERNAL VARIABLES FOR MANIPULATION
   IXA1=ICODA
   IXB1=ICODB
C
C-----SHIFT WORD CONTENTS TO RIGHT JUSTIFY.
   IXA=SHIFT(IXA1,18)
   IXB=SHIFT(IXB1,18)
C
C-----VARIABLES IXA AND IXB NOW CONTAIN THE SAME BIT PATTERN AS THE
C      ARGUMENTS ICODA AND ICODB, RIGHT JUSTIFIED, AND CAN BE COMPARED
   IF(IXA-IXB) 10,20,30
C
C      RETURN FOR ICODA .LT. ICODB
10 ICOMP=-1
   RETURN
C
C      RETURN FOR ICODA .EQ. ICODB
20 ICOMP=0
   RETURN

```

```

C
C      RETURN FOR ICODA .GT. ICODB
30 ICOMP=+1
C      RETURN

```

```

C
C      END
SUBROUTINE PCONV(NWTYP,NWVAL)

```

SUBROUTINE PCONV PERFORMS DATA UNIT CONVERSIONS OF PHYSICAL PROPERTY DATA READ IN SI TO CGS UNITS FOR HACS COMPUTATIONS. THIS ROUTINE, PERFORMING THE LIMITED CONVERSION OF SI TO CGS, WAS ADAPTED FROM SUBROUTINE OCONV USED FOR GENERAL PROPERTY FILE OUTPUT CONVERSIONS. THE ARRAY OF VALUES ON INPUT, NWVAL, IS REPLACED BY CONVERTED VALUES ON RETURN. FIELD VALUES HAVING A TYPE OF MISSING ARE NOT CONVERTED.

CONVERSION EQUATIONS ARE EVALUATED IN REVERSE FIELD SEQUENCE DUE TO DEPENDENCIES AMONG CONVERSION EQUATIONS FOR TEMPERATURE FUNCTION COEFFICIENTS. WITHIN THE LOOP ON FIELD NUMBER, THE ACTUAL CONVERSION EQUATIONS AND FACTORS TO BE APPLIED ARE SELECTED BY BRANCHING ON THE TYPE OF PHYSICAL QUANTITY STORED IN THE ARRAY XTQN AND INDEXED BY FIELD NUMBER.

AB = TEMPERATURE CONVERSION FACTOR
AG = SAME AS AB
IFLD = INTEGER INDEX FOR FIELD NUMBER
IX = INTEGER VARIABLE USED FOR INDEX IFLD+1 IN TEMPERATURE FUNCTION COEFFICIENT CONVERSIONS
KTYT = INDEX GIVING TYPE OF QUANTITY FOR FIELD IFLD USED TO REFERENCE CONVERSION DATA ARRAYS AND CONTROL THE SELECTION OF CONVERSION EQUATIONS
NWTYP = SOURCE STATUS CODES FOR FIELD VALUES READ FROM PROPERTY FILE (0=MISSING, 2=ESTIMATE, 3=EXACT)
NWVAL = ARRAY OF PROPERTY FILE FIELD VALUES
TEMP = TEMPORARY VARIABLE FOR PARTIAL SUM OF CONVERSION EQUATION TERMS
XCNV = ARRAY OF CONVERSION FACTORS USED IN SCALAR PRODUCTS OR CONVERSION EQUATIONS ACCORDING TO FIELD NUMBER AND TYPE OF PHYSICAL QUANTITY.
XTQN = INTEGER ARRAY GIVING TYPE OF PHYSICAL QUANTITY FOR EACH FIELD (1 TO 74). VALUES SHOULD NOT BE INTERPRETED TOO RIGIDLY BY PHYSICAL DEFINITION AS SOME DEGREE OF REDUNDANCY MAY BE INTENTIONALLY INCLUDED TO SIMPLIFY CONVERSION PROCESSING.

COMMON VARIABLES USED - NONE

SUBROUTINES REQUIRED - NONE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 27 SEPTEMBER 1975

```

C
C      DIMENSION NWTYP(1),NWVAL(1),XCNV(30),XTQN(74)
C      INTEGER XTQN
C      REAL NWVAL

```

```

C      DATA AB/273.15/,AG/273.15/

```

```

C      ODATA (XCNV(I),I=1,30)
1      /0.0      ,1.0      ,273.15      ,0.1      ,1000.      ,
2      273.15    ,1000.    ,1000.      ,0.1      ,1000.      ,
3      -2.30259  ,418.68   ,418.68   ,4186.8   ,4186.8   ,
4      0.001     ,1.0     ,1.0     ,2.12483  ,4186.8   ,
5      4186.8    ,4186.8   ,4186.8   ,4186.8   ,1.0     ,
6      0.01      ,1.0     ,1.0     ,1.0     ,1.0     ,/

```

```

C      ODATA (XTQN(I),I=1,74)
1      / 1, 1, 1, 2, 3, 3, 3, 4, 5, 6, 1, 7, 8, 9, 6, 6,10, 6,11, 3,
2      6, 6,12, 6,12,13, 6, 6,14, 6,14,15, 6, 6,16, 6,16, 6,17, 6,
3      17,18,19, 6, 6, 6, 6,20,21,22,23, 3, 3,24,24,24,24,24,24,
4      25,25,26,27,27,28,29,29, 1, 6,30,30, 6,30/

C
C-----INITIALIZE LOOP ON PROPERTY FIELDS. SEQUENCE IS IN REVERSE
C      ORDER TO PERMIT SEQUENTIAL CONVERSION OF TEMPERATURE FUNCTION
C      COEFFICIENTS.
C      IFLD=74
C-----RETURN HERE FOR EACH NEW FIELD, SKIP CONVERSION IF FIELD
C      VALUE IS MISSING.
10 IF(NWTYP(IFLD).EQ.0) GO TO 20
C-----BRANCH ON TYPE OF QUANTITY TO CONVERSION EQUATIONS FOR
C      EACH FIELD.
C      KTYP=XTQN(IFLD)
C      OGO TO( 20, 20, 300, 400, 400, 600, 700, 800, 900, 400,
1      1100,1200,1300,1400,1300, 400,1700,1300,1100,2000,
2      2100,2200,2300, 400, 20, 400, 20, 400, 400, 20),KTYP

C-----RETURN HERE FOLLOWING CONVERSION TO PICK UP NEXT FIELD.
C      SKIP ALHPANUMERIC FIELDS 1,2,3
20 IFLD=IFLD-1
   IF(IFLD.GT.3) GO TO 10
   RETURN

C-----CONVERSION EQUATIONS
C-----KTYP=1, FIELDS 1,2,3,11,69, NO CONVERSION REQUIRED
C 100 GO TO 20
C-----KTYP=2, FIELD 4, ALL CONVERSION FACTORS ARE UNITY
C 200 GO TO 20
C-----KTYP=3, FIELDS 5,6,7,20,52,53
300 IF(IFLD.EQ.20) GO TO 320
310 NWVAL(IFLD)=NWVAL(IFLD)/XCNV(KTYP)
320 GO TO 20
C-----KTYP=4, FIELD 8
400 NWVAL(IFLD)=NWVAL(IFLD)/XCNV(KTYP)
   GO TO 20
C-----KTYP=5, FIELD 9
C 500 GO TO 400
C-----KTYP=6, FIELDS 10,15,16,18,21,22,24,27,28,30,33,34,36,38,
C      40,44,45,46,47,70,73
600 IF(IFLD.EQ.44) GO TO 320
   IF(IFLD.NE.45) GO TO 310
   NWVAL(45)=AG*NWVAL(45)
   GO TO 20
C-----KTYP=7, FIELD 12
700 TEMP=AG*(NWVAL(13)-AG*NWVAL(14))
   GO TO 2340
C-----KTYP=8, FIELD 13
800 TEMP=2.*AG*NWVAL(14)
   GO TO 2330
C-----KTYP=9, FIELD 14
900 TEMP=0.0
   GO TO 2320
C-----KTYP=10, FIELD 17
C1000 GO TO 400
C-----KTYP=11, FIELD 19
1100 NWVAL(IFLD)=NWVAL(IFLD)-XCNV(KTYP)
   GO TO 20
C-----KTYP=12, FIELDS 23,25
1200 IF(IFLD.EQ.23) GO TO 400
1210 IX=IFLD+1
   TEMP=AG*NWVAL(IX)
   GO TO 2340
C-----KTYP=13, FIELD 26
1300 TEMP=0.0
   GO TO 2330
C-----KTYP=14, FIELDS 29,31

```


DATE - 6 APRIL 1976

```

A      = TEMPERATURE FUNCTION COEFFICIENT A
B      = TEMPERATURE FUNCTION COEFFICIENT B
C      = TEMPERATURE FUNCTION COEFFICIENT C
D      = TEMPERATURE FUNCTION COEFFICIENT D
IMN    = LOWEST SOURCE CODE FOUND ON RECALL OF COEFFICIENTS
ILO    = SOURCE CODE FOR VALUE OF TLO IN STATE FILE
IT     = SOURCE CODE FOR COMPUTED FUNCTION VALUE OBTAINED AS
        MINIMUM SOURCE CODE OF ALL VALUES ON RIGHT HAND
        SIDE OF EQUATION.
IUP    = SOURCE CODE FOR VALUE OF TUP IN STATE FILE
T      = VALUE OF TEMPERATURE TO BE USED IN COMPUTING FUNCTION
        VALUE, OBTAINED FROM AMBIENT AND BOILING TEMPERATURE
        ADJUSTED IF NECESSARY TO RANGE OF EQUATION
TLO    = LOWER LIMIT OF TEMPERATURE RANGE FOR WHICH EQUATION

```

```

C          HAS BEEN DEFINED
C          TUP      = UPPER LIMIT OF TEMPERATURE RANGE FOR WHICH EQUATION
C                   HAS BEEN DEFINED
C
COMMON/PXFER/BUFF,K,SNCOD
INTEGER SNCOD
INTEGER FLDN(74),HDR(6),YCOD,YTYP(74)
LOGICAL COEF
DIMENSION YNAM(5),YVAL(74),FREF(84)
INTEGER OREC(84),TCOD(5),BUFF(15),PTLST(29),SCOD,SCLST(28)
OEQUIVALENCE (OREC(1),YCOD,FREF(1)),(OREC(2),YNAM(1)),
1             (OREC(7),MCO), (OREC(8),SCOD), (OREC(9),TCOD(1)),
2             (OREC(14),YVAL(4)),(HDR(1),OREC(1))
ODATA (PTLST(I),I=1,29)/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,
1      1HK,1HL,1HM,1HN,1HO,1HP,1HQ,1HR,1HS,1HT,1HU,1HV,1HW,1HX,1HY,
2      1HZ,2HII,2HRR,2HSS/
ODATA (SCLST(I),I=1,28)/3HA B,3HA C,5HA B C,5HA D E,7HA D F G,
1      9HA D E F G,3HA H,5HA I J,7HA H I J,5HA K L,7HA K M N,
2      9HA K L M N,3HA O,3HA P,5HA P Q,7HA P R S,9HA P Q R S,
3      3HA T,5HA T U,5HA U V,9HA T U V W,3HA X,5HA X Y,1HZ,2HII,
4      2HRR,4HRR C,2HSS/
C
DATA      ITP/9/
ODATA (FLDN(I),I=1,74)
1      / 0, 0, 0,1002,1003,1033,1025,1034,1035,1036,
2      0,1038,1039,1040,1041,1042,1043,1044,1045,1046,
3      1047,1048,1049,1050,1051,1052,1053,1054,1055,1056,
4      1057,1058,1059,1060,1008,1061,1031,1032,1026,1027,
5      1028,1029,1010,1011,1012,1062,1063,1064,1065,1066,
6      1067,1068,1069,1070,1014,1071,1072,1073,1074,1075,
7      2033,1076,1015,2032,1077,1078,1079,1080, 0,1016,
8      1017,1018,1019,1020/
C
C          INDXX=1
C-----SET UP AUDIT OPTION
NPRRP=IPRRP
C-----WRITE STATUS MESSAGE TO USER OUTPUT
IF(NPRRP.EQ.1) CALL PAGER(0)
CALL PAGER(1)
WRITE(LP,1000) ICD
C-----REWIND TAPE AND READ HEADER RECORD.  TERMINATE IF GET END FILE.
REWIND ITP
BUFFER IN(ITP,1) (HDR(1),HDR(6))
IF(UNIT(ITP)) 30,5,5
C-----INITIAL END OF FILE ERROR CONDITION
5 CALL PAGER(2)
WRITE(LP,1010)
C-----ERROR RETURN
10 NOP=0
C-----NORMAL RETURN
20 CONTINUE
REWIND ITP
C-----RESTORE OUTPUT AUDIT
NPRRP=1
RETURN
30 CONTINUE
C-----DISPLAY FILE HEADER
30 CALL PAGER(3)
WRITE(LP,1020) HDR(5),HDR(4),HDR(3),HDR(6),HDR(2),HDR(1)
C-----RETURN HERE TO READ NEXT PHYSICAL PROPERTY RECORD
C-----TEST FOR END OF FILE.  GIVE ERROR MESSAGE AND RETURN IF

```

```

C      CHEMICAL HAS NOT BEEN FOUND.
40  BUFFER IN(ITP,1) (OREC(1),OREC(84))
    IF(UNIT(ITP)) 50,41,41
41  CALL PAGER(2)
    WRITE(LP,1030)
42  INDXX=0
    GO TO 20

C
C-----COMPARE COMPOUND RECOGNITION CODE DESIRED TO ONE JUST READ
50  CONTINUE
    IF(ICOMP(YCOD,ICD)) 40,70,60

C
C-----TAPE HAS BEEN SEARCHED PAST POSITION FOR ICD.
60  CALL PAGER(3)
    WRITE(LP,1040)
    GO TO 42

C
C-----PROPERTIES OF REQUESTED COMPOUND HAVE BEEN FOUND. PRINT DATA
C      FIELDS WHICH ARE NOT TRANSFERRED TO HACS.
70  CALL INIT(YTYP,30,5,2)
    DO 2000 I=1,74
2000  YTYP(I)=ITST(TCOD,I)
        K=LENGTH(ITP)+1
        K1=85
        DO 2020 J=4,74
            I=78-J
            K1=K1-1
            IF(YTYP(I).EQ.0) GO TO 2010
            K=K-1
            FREF(K1)=FREF(K)
            GO TO 2020
2010  FREF(K1)=0.0
2020  CONTINUE
        CALL PAGER(4)
        WRITE(LP,2030) YCOD,YNAM,YVAL(11)
2030  FORMAT (/5X,46HPHYSICAL PROPERTY DATA RETRIEVED FOR CHEMICAL ,A3/
1 9X,7HNAME = ,5A8/9X,17HSHIPPING STATE = ,A8)
        CALL INIT(ITMP,29,1,1)
        K=0
        DO 2040 I=1,29
            ITMP=ITST(MCOD,I)
            IF(ITMP.EQ.0) GO TO 2040
            K=K+1
            BUFF(K)=PTLST(I)
2040  CONTINUE
        SNCOD=SCOD
        IF(NPRRP.EQ.0) GO TO 71
        CALL PAGER(4)
        WRITE(LP,1060)
71  CONTINUE

C
C-----CONVERT FIELD VALUES FROM SI UNITS AS READ FROM PROPERTY FILE
C      TO CGS UNITS FOR INTERNAL HACS USE.
    CALL PCNV(YTYP,YVAL)

C
C-----LOOP THROUGH FIELD NUMBER LIST AND TRANSFER EXACT OR ESTIMATED
C      VALUES OF UNIT DATA FIELDS TO HACS. IF FIELD NUMBER IS ZERO,
C      VALUE IS NOT TRANSFERRED.
    DO 80 I=4,74
        IF(FLDN(I).EQ.0) GO TO 80
        IF(YTYP(I).EQ.0) GO TO 80
        CALL FSV(FLDN(I),YVAL(I),YTYP(I))
80  CONTINUE
    GO TO 20

C
C-----COMPUTATION OF TEMPERATURE FUNCTION VALUES AT AMBIENT AND
C      BOILING TEMPERATURE.
    IF(NPRRP.EQ.0) GO TO 81
    CALL PAGER(0)
    CALL PAGER(2)
    WRITE(LP,1070)
81  CONTINUE

```

```

C-----RECALL VALUES IN STATE FILE AND SAVE SOURCE CODES
    ITAM=7
    NXXX=NPRRP
    NPRRP=1
    CALL FRCL(2004,TAM,ITAM,IERR)
    NPRRP=NXXX
    ITBP=7
    CALL FRCL(FLDN(5),TBP,ITBP,IERR)

C-----COMPUTATION OF SATURATED LIQUID DENSITY.
    IF(NPRRP.EQ.0) GO TO 82
    CALL PAGER(2)
    WRITE(LP,1080)
82 CONTINUE
    IF(COEF(FLDN(12),FLDN(13),FLDN(14),0)) GO TO 90
    CALL BNDCK(1,TAM,ITAM,FLDN(16),FLDN(15))
    VAL=A+T*(B+T*C)
    CALL FSV(1004,VAL,IT)
    CALL BNDCK(2,TBP,ITBP,0,0)
    VAL=A+T*(B+T*C)
    CALL FSV(1021,VAL,IT)

C-----COMPUTATION OF LIQUID VISCOSITY.
90 IF(NPRRP.EQ.0) GO TO 91
    CALL PAGER(2)
    WRITE(LP,1090)
91 CONTINUE
    IF(COEF(FLDN(19),FLDN(20),0,0)) GO TO 100
    CALL BNDCK(1,TAM,ITAM,FLDN(22),FLDN(21))
    VAL=EXP(A+B/(T+273.15))
    CALL FSV(1006,VAL,IT)
    CALL BNDCK(2,TBP,ITBP,0,0)
    VAL=EXP(A+B/(T+273.15))
    CALL FSV(1005,VAL,IT)

C-----COMPUTATION OF LIQUID THERMAL CONDUCTIVITY.
100 IF(NPRRP.EQ.0) GO TO 101
    CALL PAGER(2)
    WRITE(LP,1100)
101 CONTINUE
    IF(COEF(FLDN(25),FLDN(26),0,0)) GO TO 110
    CALL BNDCK(1,TAM,ITAM,FLDN(28),FLDN(27))
    VAL=A+B*T
    CALL FSV(1081,VAL,IT)
    CALL BNDCK(2,TBP,ITBP,0,0)
    VAL=A+B*T
    CALL FSV(1082,VAL,IT)

C-----COMPUTATION OF LIQUID HEAT CAPACITY.
110 IF(NPRRP.EQ.0) GO TO 111
    CALL PAGER(2)
    WRITE(LP,1110)
111 CONTINUE
    IF(COEF(FLDN(31),FLDN(32),0,0)) GO TO 120
    CALL BNDCK(1,TAM,ITAM,FLDN(34),FLDN(33))
    VAL=A+B*T
    CALL FSV(1007,VAL,IT)
    CALL BNDCK(2,TBP,ITBP,0,0)
    VAL=A+B*T
    CALL FSV(1083,VAL,IT)

C-----COMPUTATION OF SOLUBILITY. NOTE THAT TEMPERATURE BOUNDS ARE
C      PRE-SPECIFIED AND NOT STORED ON THE PROPERTY FILE.
120 IF(NPRRP.EQ.0) GO TO 121
    CALL PAGER(2)
    WRITE(LP,1120)
121 CONTINUE
    IF(COEF(FLDN(41),FLDN(42),0,0)) GO TO 130
    ILO=0.0
    ILO=3
    TUP=30.0
    IUP=3

```

```

CALL BNDCK(2,TAM,ITAM,0,0)
VAL=A+B*T
CALL FSV(1084,VAL,IT)
CALL BNDCK(2,TBP,ITBP,0,0)
VAL=A+B*T
CALL FSV(1085,VAL,IT)

```

C-----COMPUTATION OF SATURATED VAPOR PRESSURE

```

130 IF(NPRRP.EQ.0) GO TO 131
CALL PAGER(2)
WRITE(LP,1130)
131 CONTINUE
IF(COEF(FLDN(43),FLDN(44),FLDN(45),0)) GO TO 140
CALL BNDCK(1,TAM,ITAM,FLDN(47),FLDN(46))
VAL=10.0*(A-B/(T+C))
CALL FSV(1086,VAL,IT)
CALL BNDCK(2,TBP,ITBP,0,0)
VAL=10.0*(A-B/(T+C))
CALL FSV(1087,VAL,IT)

```

C-----COMPUTATION OF VAPOR HEAT CAPACITY

```

140 IF(NPRRP.EQ.0) GO TO 141
CALL PAGER(2)
WRITE(LP,1140)
141 CONTINUE
IF(COEF(FLDN(48),FLDN(49),FLDN(50),FLDN(51))) GO TO 20
CALL BNDCK(1,TAM,ITAM,FLDN(53),FLDN(52))
VAL=A+T*(B+T*(C+T*D))
CALL FSV(1013,VAL,IT)
CALL BNDCK(2,TBP,ITBP,0,0)
VAL=A+T*(B+T*(C+T*D))
CALL FSV(1088,VAL,IT)
GO TO 20

```

C

```

10000FORMAT (5X,65HSTARTING SEARCH OF HACS FILE FOR PHYSICAL PROPERTIES
1 OF CHEMICAL ,A3,4H ...)
1010 FORMAT (/5X,46H****ERROR - UNABLE TO READ HACS PROPERTY FILE)
10200FORMAT (/10X,21HFILE OPENED HAS ID = ,A4,20H, VERSION NUMBER = ,
1 IS,10H, DATE = ,I6,13X,18HBACK-UP FILE ID = ,A4,20H, VERSION NU
2MBER = ,I5,10H, DATE = ,I6)
10300FORMAT (/5X,71H****ERROR - UNABLE TO FIND CHEMICAL. SEARCH TERMI
NATED BY END OF FILE)
10400FORMAT (/5X,56H****ERROR - UNABLE TO FIND CHEMICAL. SEARCH TERMI
NATED/10X,45HAFter PASSING EXPECTED ALPHABETICAL POSITION.)
10500FORMAT (/5X,46HPHYSICAL PROPERTY DATA RETRIEVED FOR CHEMICAL ,A3/
1 9X,7HNAME = ,5A8/9X,13HPATH CODES = ,8A8/9X,17HSHIPPING STATE = ,
2 A8)
10600FORMAT (/5X,46HTRANSFER OF EXACT OR ESTIMATED PROPERTY VALUES/
1 5X,30HTO HACS STATE FILE FOLLOWS .../)
10700FORMAT ( 5X,72HCOMPUTATION OF FUNCTIONS OF TEMPERATURE FOLLOW USIN
1G TEMPERATURES OF .../)
1080 FORMAT ( 5X,43HCOMPUTATION OF SATURATED LIQUID DENSITY .../)
1090 FORMAT ( 5X,35HCOMPUTATION OF LIQUID VISCOSITY .../)
1100 FORMAT ( 5X,46HCOMPUTATION OF LIQUID THERMAL CONDUCTIVITY .../)
1110 FORMAT ( 5X,39HCOMPUTATION OF LIQUID HEAT CAPACITY .../)
1120 FORMAT ( 5X,38HCOMPUTATION OF SOLUBILITY IN WATER .../)
1130 FORMAT ( 5X,43HCOMPUTATION OF SATURATED VAPOR PRESSURE .../)
1140 FORMAT ( 5X,38HCOMPUTATION OF VAPOR HEAT CAPACITY .../)
END
SUBROUTINE PTHCK(ULST,PTLST,MODNO,ISW)

```

SUBROUTINE PATH CHECK COMPARES THE RATE MODEL LETTER CODES GIVEN AS INPUT (STORED IN ULST) TO THE LIST OF VALID CODES STORED IN THE DATA ARRAY PTLST. ON RETURN, THE ARRAY MODNO CONTAINS INDEX NUMBERS FOR EACH INPUT LETTER AND ISW GIVES THE STATUS OF THE CHECKING OPERATION AS FOLLOWS-

ISW = 0 NORMAL RETURN

ISW = 1 THE INPUT LIST CONTAINS AN UNRECOGNIZABLE AND NON-BLANK RATE MODEL CODE. THE ARRAY MODNO IS ONLY PARTIALLY COMPLETED ON OUTPUT AND SHOULD NOT BE USED

ISW = 2 MODEL CODES SPECIFIED BY USER APPEAR TO BE ALL
BLANK, OR HAVE MISSING PATH CODES IN LIST

ISW = 3 MODEL CODES SPECIFIED BY THE USER ARE NOT IN THE
PROPER SEQUENCE FOR A HAZARD ASSESSMENT PATH
(EXCEPT FOR O, Z, II, RR AND SS) OR AT LEAST ONE
PATH CODE (INCLUDING O, Z, II, RR AND SS) APPEARS
MORE THAN ONCE.

ISW = 4 A MODEL CODE IS MISSING IN THE LIST GIVEN BY THE
USER

ON RETURN WITH ISW = 0,2,3 OR 4, THE ARRAY MODNO GIVES THE
INDEX NUMBER FOR EACH MODEL SPECIFIED IN THE USER LIST. IF
ISW=2, HACS MAY EITHER TERMINATE, OR SELECT DEFAULT CODES
FROM THE PROPERTY FILE. CASES WITH ISW=3 OR 4 MAY BE VALID,
BUT A WARNING MESSAGE SHOULD BE DISPLAYED. THE PROPERTY FILE
UPDATE PROGRAM DOES NOT USE THE ARRAY MODNO AND SHOULD PRO-
DUCE A FATAL ERROR ON ALL RETURNS EXCEPT ISW=0.

PATH CHECK SEQUENCE TESTS ARE SUPPRESSED IN THIS VERSION OF
SUBROUTINE PTHCK FOR RATE MODELS O, Z, II, RR AND SS WHICH
MAY APPEAR AT ANY LOCATION IN THE INPUT LIST. ADDITIONAL
LOGIC IS INCLUDED HOWEVER TO GENERATE A SEQUENCE ERROR IF
ANY OF THESE EXCEPTIONS APPEARS MORE THAN ONCE IN THE INPUT
LIST. NOTE THAT THE SEQUENCE CHECK APPLIED TO THE REMAINING
PATH CODES AUTOMATICALLY SUPPRESSES DUPLICATION.

I = INTEGER LOOP INDEX
ISW = ARGUMENT, STATUS INDICATOR ON RETURN TO CALLING
ROUTINE
J = INTEGER LOOP INDEX
LAST = TEMPORARY VARIABLE USED FOR STORAGE OF PREVIOUS
VALUE, MODNO(I-1)
MODNO = ARRAY OF NUMERIC INDICES CORRESPONDING TO USER
SPECIFIED RATE MODEL LETTERS (1 TO 26 FOR A TO Z,
27=II, 28=RR, 29=SS AND 30 FOR BLANK)
PTLST = ARGUMENT, DATA ARRAY CONTAINING ALL VALID RATE
MODEL LETTERS, A TO Z, II, RR, SS, AND BLANK.
ULST = ARGUMENT, ARRAY CONTAINS RATE MODEL LETTERS SPECIFIED
BY USER ON INPUT.

COMMON VARIABLES USED - NONE

SUBROUTINES REQUIRED - NONE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 10 JULY 1975

INTEGER MODNO(15),PTLST(30),ULST(15)

```

C-----INITIALIZE RETURN STATUS CODE, AND START LOOP ON WORDS IN USER
C INPUT LIST.
C ISW=0
C DO 20 I=1,15
C   START LOOP ON LIST OF VALID MODEL NAMES
C   DO 10 J=1,30
C   IF(ULST(I).NE.PTLST(J)) GO TO 10
C   GOT A MATCH. STORE AND SKIP UP TO OUTER LOOP
C   MODNO(I)=J
C   GO TO 20
C 10 CONTINUE
C   DID NOT FIND A MATCH. RETURN.
C   ISW=1
C   RETURN

```

```

20 CONTINUE
C-----CHECK SEQUENCE OF PATH CODES.  AT LEAST ONE NON-BLANK CODE
C      MUST APPEAR
      IF(MODNO(1).EQ.30) GO TO 60
      LAST=0
      DO 50 I=1,15
      IF(LAST.EQ.30) GO TO 30
      IF(MODNO(I).GT.LAST) GO TO 40
C-----MODEL CODE OUT OF SEQUENCE.  TEST FOR EXCEPTION, OTHERWISE
C      SET ERROR RETURN
      IF(MODNO(I).EQ.15) GO TO 24
      IF(MODNO(I).EQ.26) GO TO 24
      IF(MODNO(I).EQ.27) GO TO 24
      IF(MODNO(I).EQ.28) GO TO 24
      IF(MODNO(I).EQ.29) GO TO 24
22 ISW=3
   RETURN
C-----CHECK FOR DUPLICATION OF PATH CODES WHICH ARE EXCEPTIONS TO
C      ASCENDING SEQUENCE RULE.
24 DO 26 J=1,I
   IF(J.EQ.I) GO TO 50
   IF(MODNO(J).EQ.MODNO(I)) GO TO 22
26 CONTINUE
   GO TO 50
C
30 IF(MODNO(I).EQ.LAST) GO TO 50
   ISW=4
   RETURN
40 LAST=MODNO(I)
50 CONTINUE
   RETURN
60 ISW=2
   RETURN
   END
   SUBROUTINE RNTIO

   THIS ROUTINE IS CALLED IMMEDIATELY AFTER A USERS BASIC INPUT
   DATA DECK IS READ TO SET UP OPTIONS SELECTED FOR INPUT/OUTPUT
   CONTROL DURING AN ASSESSMENT RUN.  FIELD VALUES FROM THE
   HACS STATE FILE ARE ACCESSED AND STORED IN COMMON FOR USE
   IN EXECUTING OR SKIPPING PORTIONS OF HACS I/O DURING THE
   ASSESSMENT RUN.  EACH SET OF OPTIONS REMAINS IN EFFECT UNTIL
   THE NEXT SET OF USER INPUT CARDS ARE READ.  DIFFERENT RUN
   SET UPS MAY BE USED TO SIMPLIFY HACS OPERATIONS DEPENDING
   ON THE TYPE OF OPTIONS TO BE SELECTED.

   IR      = ERROR INDICATOR FOR DATA BASE RECALL, NOT USED
   IS      = SOURCE CODE TRACK FOR DATA BASE RECALL, NOT USED

   COMMON VARIABLES USED - ICVSL,IPRAC,IPRRP,LP,NOFF,NPRRP,WIND
   SUBROUTINES REQUIRED  - FRCL,IRCL,PAGER

   AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
           35/309A ACORN PARK,
           CAMBRIDGE, MASS., 02140
           TEL. 617-864-5770 EXT. 2813

   DATE   - 17 MAY 1976

   COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
   COMMON/IOCNT/ICVSL,IPRAC,IPRRP,NOFF,NPRRP
   COMMON/PLTCN/ANG,IBUF(4000),IFRST,IPLT,WIND
C-----AUDIT RUN TIME I/O OPTIONS SELECTED BY USER OR OBTAINED
C      BY DEFAULT

```

```

CALL PAGER(3)
NPRRP=1
C
IR=0
IS=6
WRITE(LP,2010)
2000 FORMAT (40H OUTPUT OPTIONS ARE 0=SUPPRESS, 1=SELECT)
CALL IRCL(3019,ICVSL,IS,IR)
WRITE(LP,2000)
CALL IRCL(3002,NOFF,IS,IR)
IF(NOFF.NE.0) CALL FRCL(2016,WIND,IS,IR)
WRITE(LP,2000)
CALL IRCL(3011,IPRAC,IS,IR)
WRITE(LP,2000)
201000 FORMAT(20H OUTPUT OPTIONS ARE:/
1      22H  0  SELECT ALL UNITS/
2      22H  1  SELECT CGS UNITS/
3      21H  2  SELECT SI UNITS/
4      26H  3  SELECT ENGLISH UNITS/
5      24H  4  SELECT MIXED UNITS)
CALL IRCL(3018,IPRRP,IS,IR)
RETURN
C
1000 FORMAT (/5X,30HOUTPUT CONTROL OPTIONS ARE .../)
END
OVERLAY(3,0)
PROGRAM MODA
C
C
C      MODA OBTAINS THE NECESSARY INFORMATION FOR THE EXECUTION
C      OF SUBROUTINE RLJVI. RLJVI COMPUTES THE FLOW RATE
C      FOR EITHER LIQUIDS OR GASES AS A FUNCTION OF TIME
C
COMMON/HEAD/DTE, LNCT, LNPG, LP, NPG, TITLE(10)
COMMON/C/PLTYP, XB(150)
INTEGER PLTYP
DIMENSION TIMEA(150), TEMP(150), FRT(150), PTA(150), TMGS(150), TMLS(15
10)
EQUIVALENCE (XB(1), TIMEA(1))
DATA MOD/4H  A /
CALL TRACE(0,3,0)
1 CONTINUE
IERR=0
ISC=6
C
C
C      OBTAIN FROM DATA BASE THE NECESSARY DATA ITEMS
C
CALL BEGPR(MOD)
CALL FRCL(1002,AM,ISC,IERR)
CALL FRCL(1007,CPL,ISC,IERR)
CALL FRCL(1010,AVP,ISC,IERR)
CALL FRCL(1011,BVP,ISC,IERR)
CALL FRCL(1012,CVP,ISC,IERR)
CALL FRCL(1013,CPG,ISC,IERR)
CPG=CPG/AM
CALL FRCL(1014,HVAP,ISC,IERR)
CALL FRCL(2001,VOL,ISC,IERR)
CALL FRCL(2002,HT,ISC,IERR)
CALL FRCL(2003,HH,ISC,IERR)
ISC1=8
IER1=0
CALL FRCL(2004,TO,ISC1,IER1)
IF(IER1.EQ.1) IERR=1
IF(ISC1.LT.ISC) ISC=ISC1
ISC2=8
IER2=0
CALL FRCL(2005,PTO,ISC2,IER2)
IF(IER2.EQ.1) IERR=1
IF(ISC2.LT.ISC) ISC=ISC2
IF(TO.GE.15.) CALL FRCL(1004,DL,ISC,IERR)
IF(TO.LT.15.) CALL FRCL(1021,DL,ISC,IERR)
CALL IRCL(2006,IADBT,ISC,IERR)
CALL FRCL(2007,AMSSO,ISC,IERR)

```

```

      CALL FRCL(2008,HOLED,ISC,IERR)
      CALL IRCL(2009,INC,ISC,IERR)
      CALL IRCL(3001,ITAB,IS,IR)
      CALL EPRNT(MOD,ISC,IERR,IFLAG)
      IF(IFLAG.EQ.1) GO TO 99
      IF(IFLAG.EQ.2) GO TO 1
C
C
C
C
      IF THE INITIAL PRESSURE IN THE TANK IS GIVEN, IT WILL BE USED TO
      FIND THE INITIAL TEMPERATURE OF THE CARGO. IF THE TEMPERATURE IS
      GIVEN BUT THE PRESSURE IS NOT, THE TEMPERATURE WILL BE USED. IF
      BOTH ARE NOT GIVEN, THE DEFAULT VALUE FOR TEMPERATURE WILL BE USED
      IF(ISC2.EQ.5) GO TO 3
      PTO=-1.
C
C
C
      CALL RLJVI
3 CALL RLJVI(VOL,HT,HH,HOLED,DL,PTO,AM,IADBT,TO,AMSSO,CPG,CPL,HVAP,
1AUP,BVP,CVP,INC,INS,TIMEA,TEMP,FRT,PTA,TMGS,TMLS,TVL,TIMEG,TIMEL)
C
C
C
      CALCULATE AVERAGE RATES OF GAS AND LIQUID RELEASE.
      GASRT=0.0
      RTLIQ=0.0
      IF(TIMEG.GT.0.0) GASRT=TMGS(INS)/TIMEG
      IF(TIMEL.GT.0.0) RTLIQ=TMLS(INS)/TIMEL
C
C
C
      UPDATE DATA BASE WITH OUTPUT OF RLJVI
      CALL OUTPR(MOD)
      CALL PAGER(2)
      WRITE(LP,10)
      CALL FSV(4001,TMGS(INS),4)
      CALL FSV(4047,GASRT,4)
      CALL FSV(4048,TIMEG,4)
      AVTEM=(TEMP(1)+TEMP(INS))/2.
      CALL FSV(4068,AVTEM,2)
      CALL PAGER(2)
      WRITE(LP,11)
      CALL FSV(4002,TMLS(INS),4)
      CALL FSV(4003,TVL,4)
      CALL FSV(4049,RTLIQ,4)
      CALL FSV(4050,TIMEL,4)
      IF(TMLS(INS).LT.1.) GO TO 19
      CALL PAGER(2)
      WRITE(LP,18)
      IF(TIMEL.LT.600.) CALL ISV(2029,0,4)
      IF(TIMEL.LT.600.) CALL ISV(2058,0,4)
      IF(TIMEL.LT.600.) CALL ISV(2060,0,4)
      IF(TIMEL.GE.600.) CALL ISV(2029,1,4)
      IF(TIMEL.GE.600.) CALL ISV(2058,1,4)
      IF(TIMEL.GE.600.) CALL ISV(2060,1,4)
19 IF(TMGS(INS).LT.1.) GO TO 21
      CALL PAGER(2)
      WRITE(LP,20)
      IF(TIMEG.LT.600.) CALL ISV(2061,0,2)
      IF(TIMEG.GE.600.) CALL ISV(2061,1,2)
      CALL FSV(4044,GASRT,2)
      CALL FSV(4045,TIMEG,2)
21 CALL ENDPR(MOD)
      IF(ITAB.EQ.0) GO TO 99
      DO 25 II=1,2
      CALL PAGER(0)
      CALL PAGER(4)
      WRITE(LP,12)
      CALL PAGER(1)
      WRITE(6,13)
      IF(II.NE.1) GO TO 100
      CALL PAGER(3)
      WRITE(LP,14)
100 CONTINUE
      IF(II.NE.2) GO TO 110
      CALL PAGER(3)

```

AD-A110 275

LITTLE (ARTHUR D) INC CAMBRIDGE MA
DEVELOPMENT OF A HACS USER INTERFACE MODULE.(U)
SEP 81 R G POTTS

F/G 9/2

DOT-C6-925331-A

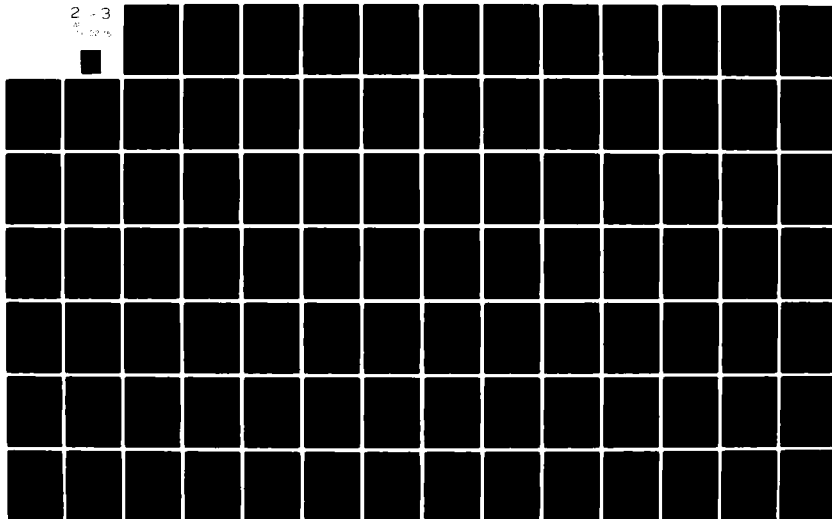
UNCLASSIFIED

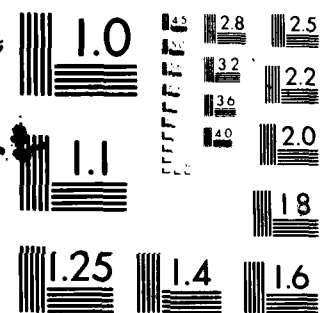
USC6-D-74-81

NL

2 - 3

1 - 10/15





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```

WRITE(LP,15)
110 CONTINUE
DO 16 I=1,INC
  IF(II.EQ.1) FLW=FRT(I)/1000.
  IF(II.EQ.1) THG=THGS(I)/1000.
  IF(II.EQ.1) TML=THLS(I)/1000.
  IF(II.EQ.1) PRES=PTA(I)/10.
  IF(II.EQ.2) FLW=FRT(I)/454.
  IF(II.EQ.2) THG=THGS(I)/454.
  IF(II.EQ.2) TML=THLS(I)/454.
  IF(II.EQ.2) PRES=PTA(I)*0.0000145077
  IF(II.EQ.1) TEM=TEMP(I)
  IF(II.EQ.2) TEM=(1.8*TEMP(I))+32.
  CALL PAGER(1)
  WRITE(LP,17) TIMEA(I),FLW,THG,TML,TEM,PRES
16 CONTINUE
25 CONTINUE
GO TO 99
10 FORMAT(1X,16HFOR GAS VENTING /)
11 FORMAT(/1X,19HFOR LIQUID VENTING /)
12 FORMAT(1H,35X,44HTABLE OF CHEMICAL VENTING PROCESS PARAMETERS//
1)
13 FORMAT(12X,11HTIME OF REL,5X,11HTOT FLWRATE,5X,11HTOT GAS REL,5X,
11HTOT LIQ REL,6X,9HTANK TEMP,7X,9HTANK PRES)
140FORMAT (15X,6H(SECS),9X,8H(KG/SEC),10X,4H(KG),12X,4H(KG),10X,
1 7H(DEG C),9X,8H(N/SQ M)//)
150FORMAT (15X,6H(SECS),9X,8H(LB/SEC),10X,4H(LB),12X,4H(LB),10X,
1 7H(DEG F),8X,10H(LB/SQ IN)//)
17 FORMAT (14X,F7.1,8X,G10.4,2(5X,G10.4),7X,F7.1,7X,G10.4)
180FORMAT(/1X,62HTHE LIQUID DISCHARGE RATE AND DURATION LEADS TO THE
1ASSUMPTION/1X,61HTHAT THE FOLLOWING SPILL DURATION INDICATORS ARE
2APPROPRIATE.)
20 FORMAT(/1X,62HIN CASE MODEL C DIRECTLY FOLLOWS, THE VAPOR DISCHARGE
*E DURATION/1X,26HINDICATOR IS ESTIMATED AS-)
99 CONTINUE
CALL TRACE(1,3,0)
END
FUNCTION RLJLC(AMASS,VOL,HT,HH,X,DL)

C
C THIS FUNCTION CALCULATES THE LIQUID LEVEL ABOVE THE HOLE
C
C THIS ROUTINE IS NOT USED SEPARATELY AND THEREFORE REQUIRES NO
C SPECIAL INPUTS
C
VOLL=(1.-X)*AMASS/DL
IF(VOLL-VOL*0.0001) 10,5,5
5 DZ=HT*VOLL/VOL-HH
GO TO 15
10 DZ=-.01
15 RLJLC=DZ
RETURN
END
SUBROUTINE RLJTC(VOL,HT,HH,X,DL,DV,PT,PA,AK,AM,IADBT,T,AMASS,AMASO
1,CPG,CPL,HVAP,AVP,BVP,CVP)

C
C THIS SUBROUTINE, CALLED BY THE VENTING RATE INTEGRATION
C SUBROUTINE, CALCULATES INSTANTANEOUS TANK CONDITIONS
C
C THIS ROUTINE IS NOT USED SEPARATELY AND THEREFORE REQUIRES NO
C SPECIAL INPUTS
C
DATA R/84836.77469/
C.....CHECK WHETHER ISOTHERMAL OR ADIABATIC
IF(IADBT)5,5,25
C.....CALCULATE TANK PRESSURE, VAPOR DENSITY, AND VAPOR WEIGHT FRACTION
C.....FOR ISOTHERMAL CASE
5 PT=AMASS*R*T/(AM*VOL)
PF=RLJVP(T,AVP,BVP,CVP)
IF(PF.LT.PA) PF=PA
IF(PT-PF)10,10,15
10 DV=AM*PT/(R*T)

```

```

      X=1.
      GO TO 55
15    PT=PF
      DV=AM*PT/(R*T)
      X=(VOL*DV*DL/AMASS-DV)/(DL-DV)
      GO TO 55
C.....SEARCH FOR THE TANK TEMPERATURE
25    TL=T
27    TH=TL
      TL=TH-10.
      CALL RLJTS(TL,TC,PC,XC,VOL,HT,HH,X,DL,DV,PT,PA,AK,AM,T,AMASS,AMASO
1,CPG,CPL,HVAP,AVP,BVP,CVP)
      IF(TL-TC)30,30,27
30    DO 45 I=1,10
      TS=(TH+TL)/2.
      CALL RLJTS(TS,TC,PC,XC,VOL,HT,HH,X,DL,DV,PT,PA,AK,AM,T,AMASS,AMASO
1,CPG,CPL,HVAP,AVP,BVP,CVP)
      IF(TS-TC)40,50,35
35    TH=TS
      GO TO 45
40    TL=TS
45    CONTINUE
50    T=TS
      PT=PC
      X=XC
55    RETURN
      END
      SUBROUTINE RLJTS(TS,TC,PC,XC,VOL,HT,HH,X,DL,DV,PT,PA,AK,AM,T,AMASS
1,AMASO,CPG,CPL,HVAP,AVP,BVP,CVP)
C
C      THIS SUBROUTINE, CALLED BY THE TANK CONDITION SUBROUTINE, USES A
C      PRESSURE-TEMPERATURE RELATIONSHIP, MASS BALANCE, AND AN
C      ENERGY BALANCE TO CALCULATE A TEMPERATURE FROM THE GIVEN ONE.
C      IF THE TWO ARE EQUAL THEN THE TEMPERATURE IS THE
C      SIMULTANEOUS SOLUTION TO THE EQUATIONS.
C
C      THIS ROUTINE IS NOT USED SEPARATELY AND THEREFORE REQUIRES NO
C      SPECIAL INPUTS
C
      DATA R/84836.73469/
C.....CHECKS TO SEE IF LIQUID IS PRESENT IN TANK
      PV=RLJVP(TS,AVP,BVP,CVP)
      PC=AMASS*R*T/(AM*VOL)
      IF(PV-PC)10,5
C.....CALCULATE TC USING IDEAL GAS EXPANSION EQUATION
5      T=((PC/PT)**((AK-1.)/AK))
      T=AM*PC/(R*TS)
      GO TO 25
C.....CALCULATES XC AND TC
      PC=PV
      DV=AM*PC/(R*TS)
      XC=(VOL*DV*DL/AMASS-DV)/(DL-DV)
      SH=AMASS*(X*CPG+(1.-X)*CPL)
      IF(RLJLQ(AMASS,VOL,HT,HH,X,DL))20,15,15
15    TC=T-(HVAP*(AMASS*XC-AMASO*X)+VOL*(PT-PC)/42680.)/SH
      GO TO 25
20    TC=T-(HVAP*(AMASO*(1.-X)-AMASS*(1.-XC))+VOL*(PT-PC)/42680.)/SH
25    RETURN
      END
      SUBROUTINE RLJVI(VOL,HT,HH,HOLED,DL,ETO,AM,IADBT,TO,AMSSO,CPG,
1CPL,HVAP,AVP,BVP,CVP,INC,IN,TIMEA,TEMP,FRT,PTA,TMGS,TNLS,TVL,TIME
2G,TIMEI)
C
C      THIS SUBROUTINE INTEGRATES NUMERICALLY THE FLOW RATE AS A
C      FUNCTION OF TIME AND PRODUCES ARRAYS OF THE TIME,
C      TEMPERATURE, INSTANTANEOUS FLOW RATE, VAPOR WEIGHT FRACTION,
C      AND TANK PRESSURE AS A FUNCTION OF THE MASS FRACTION
C      RELEASED.
C
C      *** INPUTS

```

```

VOL      VOLUME OF TANK                      CM**3
HT       HEIGHT OF TANK                      CM
HH       HEIGHT OF HOLE                      CM
HOLED    DIAMETER OF THE HOLE                CM
DL       LIQUID DENSITY                     GM/CM**3
AM       MOLECULAR WEIGHT OF THE CHEMICAL
IADBT    HEAT TRANSFER FLAG (IF POSITIVE, TANK IS ADIABATIC,
          OTHERWISE IT IS ISOTHERMAL)
PTD      INITIAL TANK PRESSURE                DYNES/CM**2.
TO       INITIAL TEMPERATURE OF TANK         DEG C
          NOTE- THE PROGRAM DOES NOT REQUIRE BOTH THE PRESSURE
          AND TEMPERATURE AS INPUT. IF THE PRESSURE IS GIVEN, IT
          WILL BE USED. IF THE TEMPERATURE IS GIVEN, THE INPUT
          FOR PRESSURE SHOULD BE SET AT SOME NEGATIVE VALUE.
AMSSO    INITIAL MASS IN TANK                GMS
CPG      HEAT CAPACITY OF VAPOR              CAL/GM-DEG C
CPL      HEAT CAPACITY OF LIQUID             CAL/GM-DEG C
HVP      HEAT OF VAPORIZATION                CAL/GM
AVP,BVP,CVP-CONSTANTS FOR A VAPOR PRESSURE EQUATION WHICH GIVES
          AN ANSWER IN MM HG. THE FORM OF THE EQUATION IS
          VAPOR PRESSURE=10.***(AVP-(BVP/(T+CVP)))
          WHERE T IS THE TEMPERATURE IN DEG C
INC      NUMBER OF MASS INCREMENTS PROGRAM IS DESIRED TO RUN FOR
          (MAXIMUM ALLOWED IS 150)

```

*** OUTPUTS

```

INS      AN INTEGER INDICATING THE NUMBER OF MASS INCREMENTS
          WHICH WERE RELEASED.
TIMEA    ARRAY OF TIMES                      SEC
TEMP     ARRAY OF TANK TEMPERATURES         DEG C
FRT      ARRAY OF FLOW RATES                 GM/SEC
PTA      ARRAY OF TANK PRESSURES             DYNES/CM**2.
TMGS     ARRAY OF TOTAL MASS GAS RELEASED    GM
TMLS     ARRAY OF TOTAL MASS LIQUID RELEASED GM
TVL      VOLUME OF LIQUID RELEASED          CM**3
TIMEG    TIME OVER WHICH GAS DISCHARGED     SECS
TIMEL    TIME OVER WHICH LIQUID DISCHARGED  SECS

```

```

ODIMENSION TIMEA(150),TEMP(150),FRT(150),PTA(150),TMLS(150),

```

```

1      TMGS(150)

```

```

DATA PA/1033.92857/

```

```

DATA R/84836.73469/

```

```

C **** PA IS EQUAL TO ONE STANDARD ATMOSPHERE IN GMF/CM**2 UNITS.

```

```

C **** R IS THE UNIVERSAL GAS CONSTANT IN GMF CM/MOLE DEG K UNITS.

```

```

C .....INITIALIZE

```

```

PI=3.14159265

```

```

RP=1.986

```

```

AK=1./((1.-(RP/(AM*CPG)))

```

```

A=PI*((HOLED/2.)**2.)

```

```

TMG=0.

```

```

TML=0.

```

```

TIMEG=0.0

```

```

TIMEL=0.0

```

```

TVL=0.

```

```

TIME=0.000

```

```

CO=.8

```

```

PT=PTD/980.7

```

```

IF(PT) 2,2,1

```

```

1 TO=(BVP/(AVP-ALOG10(0.7356*PT)))-CVP+273.16

```

```

T=TO

```

```

TO=T-273.16

```

```

GO TO 4

```

```

2 T=TO+273.16

```

```

PT=RLJVP(T,AVP,BVP,CVP)

```

```

4 DV=AM*PT/(R*T)

```

```

IF(INC-150)5,10,10

```

```

5 INCC=INC

```

```

GO TO 15

```

```

10 INCC=150

```

```

15 AMAS=AMSSO/FLOAT(INCC)

```

```

AMASS=AMSSO

```



```

IFLAG=1
RETURN
C.....CHECK WHETHER GAS FLOW IS CHOKED
10 IF(PA/PT-(2./(AK+1.))*((AK/(AK-1.)))15,15,20
C.....CALCULATE THE GAS CHOKED FLOW VENTING RATE
15 ARG=AH*G/R
CALL SRTCK(ARG,2)
ARGB=AK*(2./(AK+1.))*((AK+1.)/(AK-1.))
CALL SRTCK(ARGB,3)
ARGC=T
CALL SRTCK(ARGC,4)
W=CO*SQRT(ARG)*SQRT(ARGB)*PT*A/SQRT(ARGC)
IFLAG=2
RETURN
C.....CALCULATE THE GAS NON-CHOKED FLOW VENTING RATE
20 B=PA/PT
IF(B-1.) 21,22,22
21 ARG=B**((2./AK)*AK/(AK-1.))*(1.-B**((AK-1.)/AK))/(1.-B)
CALL SRTCK(ARG,5)
ARGB=2.*G*(PT-PA)*DV
CALL SRTCK(ARGB,6)
W=CO*SQRT(ARG)*A*SQRT(ARGB)
GO TO 23
22 W=0.
23 IFLAG=2
25 RETURN
END
SUBROUTINE SRTCK(ARG,N)
C
C SPECIAL SQUARE ROOT ARGUMENT TEST ROUTINE INSERTED BY
C R.G. POTTS ON 7 APRIL 1978. IF ARGUMENT IS LESS THAN
C 0.0, THE ROUTINE GENERATES A WARNING MESSAGE THEN
C SUBSTITUTES A VALUE OF 0.0 ON RETURN. THE INTEGER
C N IS A LOCATION REFERENCE TO THE CALLING PROGRAM.
C REFERENCES 1 TO 6 ARE USED BY SUBROUTINE VENTR.
C
COMMON/HEAD/DTE,LNCT,LNPG,LP,NPG,TITLE(10)
C
IF(ARG.GE.0.0) RETURN
CALL PAGER(2)
WRITE(LP,1000) ARG,N
ARG=0.0
RETURN
10000FORMAT (5X,38H*****WARNING, SQUARE ROOT ARGUMENT OF ,G13.4/
1 10X,36H WAS SET TO 0.0 AT PROGRAM LOCATION ,I2)
END
OVERLAY(4,0)
PROGRAM DV4

DV4 EXECUTES THE FOLLOWING INTER-RELATED GROUP OF FLAME SIZE
AND THERMAL RADIATION RATE MODELS -

RATE MODEL = B INDEX = 2
              E 5
              H 8
              L 12
              Q 17
              U 21

H = DUMMY INTERNAL VARIABLE USED TO TRANSFER VALUE OF
   FIELD 4018 TO FIELD 4006
IR = ERROR VALUE RETURNED BY FRCL ROUTINE
IS = VALUE OF SOURCE CODE UPDATED IN FRCL, NOT USED IN
     PROGRAM DV4

COMMON VARIABLES USED - MODNO
SUBROUTINES REQUIRED - FRCL,FSV,MODB1,MODB2,MODE1,MODE2,MODH,
                     MODL,MODQ,MODU,TRACE
AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
          35/309A ACORN PARK,

```

```

C                                     CAMBRIDGE, MASS., 02140
C                                     TEL. 617-864-5770 EXT. 2813
C
C      DATE   - 8 JANUARY 1976
C
C      COMMON/OVCNT/MODNO,OVLST(29),SGLST(29)
C      INTEGER      OVLST,SGLST
C
C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN BRANCH ON MODEL
C      INDEX NUMBER
C      CALL TRACE(0,4,0)
C
C-----SELECT MODEL B
C      IF(MODNO.NE.2) GO TO 10
C      CALL MODB1
C      CALL MODB2
C      GO TO 100
C
C-----SELECT MODEL E
C      10 IF(MODNO.NE.5) GO TO 20
C      CALL MODE1
C      CALL MODE2
C      CALL FRCL(4018,H,IS,IR)
C      CALL FSU(4006,H,6)
C      CALL MODB2
C      GO TO 100
C
C-----SELECT MODEL H
C      20 IF(MODNO.NE.8) GO TO 30
C      CALL MODH
C      GO TO 100
C
C-----SELECT MODEL L
C      30 IF(MODNO.NE.12) GO TO 40
C      CALL MODL
C      GO TO 100
C
C-----SELECT MODEL Q
C      40 IF(MODNO.NE.17) GO TO 50
C      CALL MODQ
C      GO TO 100
C
C-----SELECT MODEL U
C      50 IF(MODNO.NE.21) GO TO 100
C      CALL MODU
C
C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN RETURN TO MAIN
C      HACS CONTROL
C      100 CALL TRACE(1,4,0)
C
C      END
C      FUNCTION ARSIN(A)
C
C      ARSIN COMPUTES THE ARC SINE OF AN ANGLE WHOSE SINE IS GIVEN
C      ARGUMENTS --
C      A = SINE OF THE ANGLE
C      ARSIN = THE ANGLE WHOSE SINE IS A
C
C      B=SQRT(1.-A**2)
C      IF(B.NE.0.) GO TO 1
C      ARSIN=0.
C      CALL PAGER(2)
C      WRITE(6,100)
C      RETURN
C      1 ARSIN=ATAN(A/B)
C      RETURN
C      100 FORMAT(23H ERROR IN ARSIN ROUTINE/)
C      END
C      SUBROUTINE CVERT(D,IOUT)
C
C      THIS ROUTINE CONVERTS UNITS OF INPUT AND OUTPUT DATA TO PROPER

```

```

C      UNITS FOR USE BY ROUTINE PROTNK.
C
C *** INPUTS AND OUTPUTS ***
C
C      D      ARRAY OF INPUT AND OUTPUT DATA FROM ROUTINE PROTNK
C              ( SEE HEADING OF PROTNK FOR DEFINITIONS)
C      IOUT    INDICATOR FLAG.  WHEN IOUT=1, CVERT CONVERTS DATA
C              FROM CGS TO ENG UNITS.  WHEN IOUT=2, IT CONVERTS
C              FROM ENG UNITS TO CGS UNITS.
C
C      DIMENSION D(26),C(5)
C
C      DIVISION BY C(1) CONVERTS UNITS FROM CM TO FEET
C      C(1)=30.48
C      DIVISION BY C(2) CONVERTS UNITS FROM CM TO INCHES
C      C(2)=2.54
C      DIVISION BY C(3) CONVERTS UNITS FROM DYNES/CM**2 TO PSI
C      C(3)=68975.72
C      DIVISION BY C(4) CONVERTS UNITS FROM CAL/SEC-CM2 TO BTU/HR-FT2
C      C(4)=0.0000753474
C      DIVISION BY C(5) CONVERTS UNITS FROM CAL/SEC-CM-DEG C TO
C      BTU/HR-FT-DEG F
C      C(5)=0.0041339
C
C      IF(IOUT.EQ.1) GO TO 5
C      DO 3 I=1,5
C      3 C(I)=1./C(I)
C      5 D(1)=D(1)/C(1)
C      D(2)=D(2)/C(2)
C      D(4)=D(4)/C(3)
C      D(5)=D(5)/C(4)
C      DO 10 I=6,10
C      10 D(I)=D(I)/C(5)
C      DO 20 I=16,20
C      20 D(I)=D(I)/C(3)
C      IF(IOUT.EQ.1) GO TO 30
C      D(22)=D(22)/C(3)
C      D(23)=(D(23)-32.)*(5./9.)
C      D(24)=D(24)/C(3)
C      D(25)=(D(25)-32.)*(5./9.)
C      D(26)=D(26)*60.
C      30 RETURN
C      END
C      SUBROUTINE FLJET(HOLED,XMOL,TADIA,ALPHA,AFR,XLEN,DE)
C
C *** THIS ROUTINE CALCULATES THE FLAME LENGTH AND DIAMETER (OF AN
C      EQUIVALENT CYLINDRICAL FLAME) FOR A NON-PREMIXED TURBULENT
C      FLAME BASED ON HOTTELS ANALYSIS OF TURBULENT FLAMES.  THE
C      FLAME PARAMETERS ARE FAIRLY INSENSITIVE TO THE PRESSURE IN THE
C      TANK.
C      *****
C      ***** INPUT ARGUMENTS *****
C      *** HOLED  DIAMETER OF HOLE IN THE TANK      CMS
C      *** XMOL   MOLECULAR WEIGHT OF FUEL          GMS/MOL
C      *** TADIA  ADIABATIC FLAME TEMPERATURE      DEG C
C      *** ALPHA  MOLAR RATIO OF REACTANTS TO PRODUCTS
C      *** AFR    STOICHIOMETRIC AIR FUEL RATIO    GM OF AIR/GM OF FUEL
C              FOR STOICHIOMETRIC COMBUSTION
C      ***** OUT PUT ARGUMENTS *****
C      *** XLEN   LENGTH OF FLAME                  CMS
C      *** DE     DIAMETER OF EQUIVALENT CYLINDRICAL FLAME (FOR USE IN THERMAL RADIATION MODELS).  CMS
C
C      TF=TADIA+273.
C      CT=1./(1.+AFR*XMOL/28.9)
C      RHS=(5.3/CT)*SQRT((CT+(1.-CT)*28.9/XMOL)*(TF/(ALPHA*300.)))
C *** 300 DEG KELVIN IS USED FOR AMBIENT TEMPERATURE.
C *** SEMI ANGLE OF JET IS ASSUMED TO BE 5.4 DEGREES.
C      XLEN=HOLED*RHS
C      DE=HOLED+XLEN/10.6
C      RETURN
C      END
C      SUBROUTINE FLMAN(D,PG,UN,ALPHA)

```

```

C
C*****
C      THIS SUBROUTINE CALCULATES THE FLAME ANGLE WITH THE VERTICAL
C      USING THE MODEL OF SLIEPCEVICH, WELKER, HUFFMAN, AND PIPKIN,
C*****
C**** INPUT ARGUMENTS ****
C      D      POOL DIAMETER          CMS
C      PG     FUEL VAPOR DENSITY AT THE TEMP OF LIQUID.  GM/CM**3
C      UW     WIND VELOCITY          CM/SEC
C*****
C**** OUTPUT ARGUMENT ****
C      ALPHA  FLAME ANGLE WITH RESPECT TO THE VERTICAL  RADIANS
C*****
C
C      ALPHA=0.
C      IF(UW.LE.0.) RETURN
C      TC=3.3*((D*UW/0.15)**0.07)*((UW**2/(980.*D))**0.8)*((PG/1.2E-3)**
C      1*(-0.6))
C
C *** 0.15 IS THE KINEMATIC VISCOSITY OF AIR IN CM**2/SEC
C *** 1.2E-3 IS THE DENSITY OF AIR IN GM/CM**3
C      ALPHA=ARCSIN((-1.+SQRT(1.+4.*TC**2))/(2.*TC))
C      RETURN
C      END
C      SUBROUTINE FLMHT(DIA,D,R,H)
C
C      THIS ROUTINE CALCULATES THE FLAME HEIGHT (CM) OF A POOL FIRE
C      USING THOMAS=S MODEL.
C
C      ARGUMENTS
C      DIA    POOL DIAMETER,CM
C      D      DENSITY OF LIQUID FUEL, GM/CM**3
C      R      THE BURNING RATE IN CM/SEC
C      H      FLAME HEIGHT,CM
C
C      H=DIA**42.*(R*D/(1.2E-3*SQRT(980.*DIA)))**0.61
C      RETURN
C      END
C      SUBROUTINE JHHRF(DIA,H,S,ALPHA,T,HF)
C
C      THIS ROUTINE CALCULATES THE MAXIMUM RADIATION FLUX POSSIBLE FROM
C      A FLAME OF GIVEN CHARACTERISTICS AT A GIVEN DISTANCE FROM THE
C      CENTER OF THE BASE OF THE FLAME
C      SOLAR INSULATION=300 BTU/HR-SQ FT.
C
C      **** INPUT ARGUMENTS
C      T      FLAME TEMPERATURE,DEGREES C
C      DIA    DIAMETER OF THE BURNING POOL,CM
C      H      HEIGHT OF THE FLAME,CM
C      S      DISTANCE OF THE OBSERVER FROM CENTER OF BASE OF FLAME,CM
C      ALPHA  ANGLE OF PLUME FROM VERTICAL,RADIANS
C
C      *** OUTPUT ARGUMENTS
C      HF     RADIATION FLUX,CAL/SEC-SQ CM
C
C      PI=3.14159265
C      S=S/(2.54*12.)
C      ST=T
C      H=H/(2.54*12.)
C      T=460.+((9./5.)*T)+32.
C      DIA=DIA/(2.54*12.)
C      SB=.1713E-08
C      EM=1.
C      TR=1.
C      EMPWR=SB*TR*EM*(T**4.)
C      CALL SUEIW(ALPHA,H,S,DIA,FMAX)
C      HF=(FMAX*EMPWR)+300.
C      HF=HF*(252./(3600.*144.*2.54*2.54))
C      T=ST
C      DIA=DIA*(2.54*12.)
C      S=S*(2.54*12.)

```

```

H=H*2.54*12.
RETURN
END
SUBROUTINE MODB1

C
C
C
C
SUBROUTINE MODB1 IS A MODULE OF MODEL B. IT OBTAINS THE NECESSARY
DATA TO CALL ROUTINE FLJET, THE ROUTINE WHICH COMPUTES THE FLAME
LENGTH AND EQUIVALENT DIAMETER WHEN A GAS VENTING FROM A TANK IS
IGNITED AT ITS SOURCE
DATA MOD/4H B1 /
1 CONTINUE
IR=0
IS=6

C
C
C
OBTAIN NECESSARY DATA ITEMS

CALL BEGPR(MOD)
CALL FRCL(2008,HOLED,IS,IR)
CALL FRCL(1002,XMOL,IS,IR)
CALL FRCL(1016,TADIA,IS,IR)
CALL FRCL(1017,ALPHA,IS,IR)
CALL FRCL(1018,AFR,IS,IR)
CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 1

C
C
C
CALL FLJET AND UPDATE DATA BASE WITH OUTPUT FROM FLJET

CALL FLJET(HOLED,XMOL,TADIA,ALPHA,AFR,XLEN,DE)
CALL OUTPR(MOD)
CALL FSV(4006,XLEN,4)
CALL FSV(4007,DE,4)

C
C
C
AUDIT

CALL ENDPR(MOD)
99 RETURN
END
SUBROUTINE MODB2

C
C
C
SUBROUTINE MODB2 IS PART OF MODELS B AND E. IT CALCULATES SAFE
SEPARATION DISTANCES FROM FLAMES AND THE RADIATION FLUX VS
DISTANCE FROM THE FLAME USING SUBROUTINE JHHRF

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
DIMENSION AS(20),AH(20),SAFSP(4),HTFLX(4),ASAV(20)
DIMENSION PTITL(6),XTITL(6),XTITL1(6),YTITL(6)
EQUIVALENCE (XBX(41),AS(1)),(XBX(1),AH(1))
EQUIVALENCE (XBX(21),ASAV(1)),(XBX(61),SAFSP(1))
ODATA (PTITL (I),I=1,6)/BHRADIATIO,8HN FLUX V,8HS DISTAN,
18HCE - MOD,8HEL R ,1H /
ODATA (XTITL (I),I=1,6)/8HDISTANCE,8H FROM FL,8HAME CENT,
18HER.....,8H.....,8H(METERS)/
ODATA (XTITL1(I),I=1,6)/8HDISTANCE,8H FROM FL,8HAME CENT,
18HER.....,8H.....,8H..(FEET)/
ODATA (YTITL (I),I=1,6)/BHRADIATIO,8HN FLUX ,8H (CAL/CM,
18H2-S) ,1H ,1H /
DATA MOD/4H B2 /
DATA HTFLX(1),HTFLX(2),HTFLX(3)/0.753,0.113,0.0339/
100 CONTINUE
IR=0
IS=6
LP=6

C
C
C
OBTAIN NECESSARY DATA ITEMS FOR JHHRF

CALL BEGPR(MOD)
CALL IRCL(2082,ITKHT,IS,IR)
IF(ITKHT.GT.0) CALL FRCL(2083,TKDIS,IS,IR)
CALL FRCL(1019,T,IS,IR)

```

```

CALL FRCL(4006,H,IS,IR)
CALL FRCL(4007,DIA,IS,IR)
CALL FRCL(4008,ALPHA,IS,IR)
IF(ITKHT.EQ.1) GO TO 101
IERR=0
CALL FRCL(2010,HTFLX(4),IS,IR)
IF(HTFLX(4).GT.0.0.AND.HTFLX(4).LE.0.0226) IERR=1
CALL IRCL(3003,IB2SF,IS,IR)
101 CONTINUE
CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 100
IF(DIA.LE.0.0) CALL PAGER(4)
IF(DIA.LE.0.0) WRITE(LP,96)
IF(DIA.LE.0.0) GO TO 99
IF(ITKHT.EQ.1) GO TO 97

CCCCC
ITERATION TO FIND SAFE SEPARATION DISTANCES
SAFE SEPARATION DISTANCES CALCULATED ARE ONLY ACCURATE TO
WITHIN 10 FEET.
THE MINIMUM DISTANCE WHICH CAN BE AN OUTPUT IS 5 FEET.

SAFSP(4)=0.0
DISMN=DIA/2.
IF(IERR.EQ.1) HTFLX(4)=0.0
DO 50 ITER=1,4
S=DISMN+(10.*2.54*12.)
ISTP=0
IF(HTFLX(ITER).EQ.0.0) GO TO 50
15 CALL JHHRF(DIA,H,S,ALPHA,T,HF)
IF(HF-HTFLX(ITER)) 20,25,30
20 S=S-(10.*2.54*12.)
IF(S.LE.DISMN) S=DISMN+(5.*2.54*12.)
ISTP=1
IF(S.EQ.(DISMN+(5.*2.54*12.)))GO TO 25
GO TO 15
25 SAFSP(ITER)=S-DISMN
GO TO 50
30 IF(ISTP.EQ.1) GO TO 25
S=S+(100.*12.*2.54)
GO TO 15
50 CONTINUE

CCCCC
UPDATE DATA BASE WITH OUTPUT FROM JHHRF

CALL OUTPR(MOD)

SAFE SEPARATION DISTANCES ARE MEASURED FROM THE EDGE OF
THE FLAME.

CALL FSV(4009,SAFSP(1),4)
CALL FSV(4015,SAFSP(2),4)
CALL FSV(4017,SAFSP(3),4)
IF(SAFSP(4).NE.0.0) CALL FSV(4034,SAFSP(4),4)
CALL PAGER(2)
WRITE(LP,51)
IF(IERR.NE.1) GO TO 55
CALL PAGER(2)
WRITE(6,40)
55 CONTINUE
CALL ENDPR(MOD)
IF(IB2SF.NE.1) GO TO 1

CCCCC
CALL JHHRF TO OBTAIN RADIATION FLUX AS A FUNCTION OF
REQUESTED DISTANCE

SMX=DISMN+SAFSP(3)
SMN=DISMN+SAFSP(1)
AX=(SMX-SMN)/19.
AS(1)=SMN
CALL JHHRF(DIA,H,AS(1),ALPHA,T,AH(1))
DO 10 I=2,20

```

```

      AS(I)=AS(I-1)+AX
      CALL JHHRF(DIA,H,AS(I),ALPHA,T,AH(I))
C
C      WRITE PLOT FILE INFORMATION
10  CONTINUE
    DO 35 I=1,20
35  ASAV(I)=AS(I)/100.
    DIV=1./3.281
    CALL PLTLP(PTITL,ASAV,AH,20,XTITL,YTITL,1,DIV,XTITL1)
C
C-----SET UP OFF-LINE PLOT
    PLTYP=2
C
    1 CONTINUE
    IF(ITKHT.EQ.0) GO TO 99
97  CALL JHHRF(DIA,H,TKDIS,ALPHA,T,RDFLX)
    CALL PAGER(5)
    WRITE(6,98)
    CALL FSV(2066,RDFLX,4)
    CALL MODB3
99  RETURN
400 FORMAT (/1X,61H THE USER GIVEN RADIATION FLUX IS LESS THAN THAT FRO
1M THE SUN./1X,51H THE MODEL IS THEREFORE NOT EXECUTED FOR THIS VALU
2E.)
51 FORMAT(5X,47H*** SAFE SEPARATION DISTANCES ARE MEASURED FROM/9X,27
1H THE OUTER EDGE OF THE FLAME)
960 FORMAT(/5X,74H*** MODEL CANNOT BE EXECUTED FOR A ZERO OR NEGATIVE
1 POOL DIAMETER. ****/5X,34H*** EXECUTION IS TERMINATED. ****/)
98  FORMAT(/61H EXECUTION OF THE COMPRESSED LIQUEFIED GAS TANK HEATING
1 MODEL/62H HAS BEEN REQUESTED. THE FOLLOWING OBTAINS THE NECESSAR
2Y DATA/25H AND EXECUTES THIS MODEL./)
    END
    SUBROUTINE MODB3
C
C      THIS SUBROUTINE OBTAINS THE DATA NECESSARY FOR AND EXECUTES THE
C      COMPRESSED LIQUEFIED GAS TANK HEATING MODEL. INPUTS AND OUTPUTS
C      ARE DESCRIBED IN THE HEADING FOR SUBROUTINE PROTANK.
C
    DIMENSION D(26)
    DATA MOD/4H B3 /
    1 CONTINUE
    LP=6
    CALL BEGPR(MOD)
    IS=6
    IR=0
    CALL FRCL(1010,AVP,IS,IR)
    CALL FRCL(1011,BVP,IS,IR)
    CALL FRCL(1012,CVP,IS,IR)
    DO 10 I=1,20
    II=I+2061
10  CALL FRCL(II,D(I),IS,IR)
    CALL EPRNT(MOD,IS,IR,IL)
    IF(IL.EQ.1) GO TO 99
    IF(IL.EQ.2) GO TO 1
    CALL PROTANK(D,AVP,BVP,CVP)
    II=D(21)
    CALL OUTPR(MOD)
    IF(II.NE.1) GO TO 5
    CALL PAGER(3)
    WRITE(LP,40)
    CALL FSV(4051,D(22),4)
    CALL FSV(4052,D(23),4)
    5 IF(II.NE.2) GO TO 6
    CALL PAGER(3)
    WRITE(LP,30)
    CALL FSV(4053,D(24),4)
    CALL FSV(4054,D(25),4)
    6 IF(II.NE.3) GO TO 7
    CALL PAGER(3)
    WRITE(LP,20)
    7 IF(II.EQ.1.OR.II.EQ.2) CALL FSV(4055,D(26),4)
    CALL ENDPR(MOD)

```


C
C
C
C

AND TRANSFERS THE VALUE FOR THE HEIGHT OF THE FLAME (COMPUTED
IN MODE1) FROM FIELD NUMBER 4018 TO 4006.

```

CALL PAGER(2)
WRITE(6,100)
CALL FRCL(4003,VOL,IS,IR)
D=2.*VOL**(.1./3.)
CALL PAGER(2)
WRITE(6,102)
CALL FSV(4007,D,4)
CALL MODE1
CALL MODE2
CALL FRCL(4018,H,IS,IR)
CALL FSV(4006,H,6)
CALL MODB2
CALL PAGER(2)
WRITE(6,101)
RETURN
100 FORMAT(82H MODEL H WILL BE REPLACED BY THE EXECUTION OF E1,E2, AND
1B2 - EXECUTION PROCEEDING./)
101 FORMAT(18H MODEL H EXECUTED./)
102 FORMAT(123H MODEL H ASSUMES THAT THE DIAMETER OF THE FLAME IS TWIC
*E THE VOLUME OF THE LIQUID DISCHARGED TO THE 1/3 POWER. THEREFORE.
*./)
END
SUBROUTINE MODL

```

C
C
C
C
C
C
C
C

SUBROUTINE MODL CALLS THE THERMAL RADIATION ESTIMATION ROUTINES
OF MODEL E FOR SPILLS OF SOLUBLE, FLAMMABLE OR COMBUSTIBLE LIQUIDS
WHICH HAVE A BOILING POINT LESS THAN THE AMBIENT. IN DOING SO,
IT ESTIMATES THE DIAMETER OF THE BASE OF THE FLAME AND
TRANSFERS THE VALUE FOR THE HEIGHT OF THE FLAME (COMPUTED IN
MODE1) FROM FIELD NUMBER 4018 TO 4006.

```

CALL PAGER(2)
WRITE(6,100)
CALL FRCL(1021,DENL,IS,IR)
CALL FRCL(2021,H,IS,IR)
CALL FRCL(4003,VOL,IS,IR)
D=2.*VOL**0.333333
IF(H.LT.304.8.AND.DENL.LT.1.0) D=24.*VOL**0.333333
CALL PAGER(2)
WRITE(6,102)
CALL FSV(4007,D,2)
CALL MODE1
CALL MODE2
CALL FRCL(4018,H,IS,IR)
CALL FSV(4006,H,6)
CALL MODB2
CALL PAGER(2)
WRITE(6,101)
RETURN
100 FORMAT(85H MODEL L IS REPLACED BY THE EXECUTION OF MODELS E1,E2, A
1ND B2 - EXECUTION PROCEEDING./)
101 FORMAT(18H MODEL L EXECUTED./)
102 FORMAT(5X,55HMODEL L ESTIMATES THE BURNING POOL DIAMETER AS BEING.
1./)
END
SUBROUTINE MODQ

```

C
C
C
C
C
C
C
C

SUBROUTINE MODQ CALLS THE THERMAL RADIATION ESTIMATION ROUTINES
OF MODEL E FOR SPILLS OF SOLUBLE, FLAMMABLE OR COMBUSTIBLE LIQUIDS.
IN DOING SO, IT ESTIMATES THE DIAMETER OF THE BASE OF FLAME
AND TRANSFERS THE VALUE FOR THE HEIGHT OF THE FLAME (COMPUTED
IN MODE1) FROM FIELD NUMBER 4018 TO 4006.

```

CALL PAGER(2)
WRITE(6,100)
CALL FRCL(4003,VOL,IS,IR)

```

```

D=2.*VOL**(1./3.)
CALL PAGER(3)
WRITE(6,102)
CALL FSV(4007,D,4)
CALL MODE1
CALL MODE2
CALL FRCL(4018,H,IS,IR)
CALL FSV(4006,H,6)
CALL MODB2
CALL PAGER(2)
WRITE(6,101)
RETURN
100 FORMAT(85H MODEL Q IS REPLACED BY THE EXECUTION OF MODELS E1,E2, A
1ND B2 - EXECUTION PROCEEDING./)
101 FORMAT(18H MODEL Q EXECUTED./)
102 FORMAT(67H MODEL Q ASSUMES THAT THE DIAMETER OF THE FLAME IS TWICE
* THE VOLUME/ 56H OF THE LIQUID DISCHARGED TO THE 1/3 POWER, THEREF
*ORE.../)
END
SUBROUTINE MODU

```

CCCCC
SUBROUTINE MODU CALLS THE THERMAL RADIATION ESTIMATION ROUTINES OF MODEL E FOR SPILLS OF LIGHTER-THAN-WATER, INSOLUBLE, FLAMMABLE OR COMBUSTIBLE LIQUIDS. IN DOING SO, IT TRANSFERS THE VALUE FOR THE HEIGHT OF THE FLAME (COMPUTED IN MODE1) FROM FIELD NUMBER 4018 TO 4006. THE POOL SIZE USED IS THAT COMPUTED IN MODT.

```

CALL PAGER(2)
WRITE(6,100)
CALL MODE1
CALL MODE2
CALL FRCL(4018,H,IS,IR)
CALL FSV(4006,H,6)
CALL MODB2
CALL PAGER(2)
WRITE(6,101)
RETURN
100 FORMAT(77H MODEL U IS REPLACED BY EXECUTING MODELS E1,E2 AND B2 -
1EXECUTION PROCEEDING./)
101 FORMAT(18H MODEL U EXECUTED./)
END
SUBROUTINE PROTNK(D,AVP,BVP,CVP)

```

CCCCC

SUBROUTINE PROTNK EVALUATES THE RESPONSE OF THE WALL OF A CARGO TANK CONTAINING A COMPRESSED LIQUEFIED GAS (NOT ONE WHICH HAS BEEN LIQUEFIED BY REFRIGERATION) WHEN SUBJECTED TO HEATING BY AN EXTERNAL FIRE. OUTPUT INCLUDES THE TIME TO RUPTURE, MECHANICAL STRESS IN THE WALL AT THE TIME OF RUPTURE, AND THE WALL TEMPERATURE. NOTE- THE ROUTINE REQUIRES DETAILED SPECIFICATION OF THE PHYSICAL PROPERTIES OF THE WALL. DEFAULT VALUES CONTAINED IN HACS ARE TYPICAL OF THOSE FOR A PROPYLENE CARGO TANK.

*** INPUTS ***

CCCCC
D(1) INTERNAL TANK DIAMETER CM
D(2) WALL THICKNESS CH
D(3) FRACTION OF TANK VOLUME CONTAINING VAPOR NON-DIM
D(4) RELIEF VALVE SETTING (GAUGE PRESSURE) DYNES/CM2
D(5) HEAT FLUX AT OUTSIDE SURFACE OF TANK CAL/CM2-SEC
D(6) WALL THERMAL CONDUCTIVITY AT 0 DEG F CAL/CM-S-DEG C
D(7) WALL THERMAL CONDUCTIVITY AT 400 DEG F CAL/CM-S-DEG C
D(8) WALL THERMAL CONDUCTIVITY AT 800 DEG F CAL/CM-S-DEG C
D(9) WALL THERMAL CONDUCTIVITY AT 1200 DEG F CAL/CM-S-DEG C
D(10) WALL THERMAL CONDUCTIVITY AT 1600 DEG F CAL/CM-S-DEG C
D(11) SPECIFIC HEAT OF WALL AT 0 DEG F CAL/GM-DEG C
D(12) SPECIFIC HEAT OF WALL AT 400 DEG F CAL/GM-DEG C
D(13) SPECIFIC HEAT OF WALL AT 800 DEG F CAL/GM-DEG C
D(14) SPECIFIC HEAT OF WALL AT 1200 DEG F CAL/GM-DEG C
D(15) SPECIFIC HEAT OF WALL AT 1600 DEG F CAL/GM-DEG C
D(16) ULTIMATE TENSILE STRENGTH OF WALL AT 0 DEG F DYNES/CM2
D(17) ULTIMATE TENSILE STRENGTH OF WALL AT 400 DEG F DYNES/CM2

```

C      D(18)    ULTIMATE TENSILE STRENGTH OF WALL AT 800 DEG F DYNES/CM2
C      D(19)    ULTIMATE TENSILE STRENGTH OF WALL AT 1200 DEG F DYNES/CM2
C      D(20)    ULTIMATE TENSILE STRENGTH OF WALL AT 1600 DEG F DYNES/CM2
C      AVP, BVP, AND CVP ARE CORRELATION CONSTANTS FOR THE VAPOR
C      PRESSURE EQUATION WHICH GIVES AN ANSWER IN UNITS
C      OF MM HG.
C
C      *** OUTPUTS ***
C
C      D(21)    FAILURE PARAMETER FLAG. D(21)=1.0 MEANS FAILURE OCCURS
C              AT OUTSIDE OF WALL, =2.0 MEANS ITS AT INSIDE OF WALL,
C              =3.0 MEANS THAT FAILURE DOES NOT OCCUR-THE TANK WALL CAN
C              WITHSTAND STRESSES AND IS IN A STEADY-STATE CONDITION.
C      D(22)    FAILURE STRESS AT OUTSIDE OF WALL DYNES/CM2
C      D(23)    TEMPERATURE AT OUTSIDE OF WALL DEG C
C      D(24)    FAILURE STRESS AT INSIDE OF WALL DYNES/CM2
C      D(25)    TEMPERATURE AT INSIDE OF WALL DEG C
C      D(26)    IF D(22)=1.0 OR 2.0, THIS GIVES THE ELAPSED TIME
C              FROM START OF FIRE UNTIL FAILURE OCCURS SECS
C      *****
C      REAL L1,L2,L3,L4,L5,K12,K23,K1,K2,K3,MCV1,MCV2,L6
C      DIMENSION X(25),D(26)
C      DATA PI,SBC/3.1415927,.1713E-8/
C      DATA DT/1.666667E-3/
C      IO=5
C      IOUT=1
C      CALL CVERT(D,IOUT)
C      DO 8 IXD=1,4
C      8 X(IXD)=D(IXD)
C      DO 9 IXD=6,20
C      9 X(IXD-1)=D(IXD)
C      DO 10 IXD=21,26
C      10 X(IXD-1)=D(IXD)
C
C      CALCULATE SATURATION TEMPERATURE AT RELIEF VALVE SETTING
C      PSAT=(X(4)+14.696)/14.696
C      PMH=PSAT*760.
C      TSAT=(BVP/(AVP-ALOG10(PMH)))-CVP
C      TSAT=(1.8*TSAT)+492.
C      TSAT4=TSAT**4
C      T3=TSAT+30.
C
C      SOLVE FOR COEFFICIENTS OF A CURVE FIT OF THE PHYSICAL PROPERTIES
C      OF THE WALL
C      A0=X( 7)
C      A4=4.*( .5*(X( 5)+X( 9))-2.*(X( 6)+X( 8))+3.*A0)/3.
C      A2=2.*(X( 6)+X( 8))-2.*A0-.125*A4
C      A3=4.*(X( 9)-2.*X( 8)+A0-.5*A2-.875*A4)/3.
C      A1=X( 9)-A0-A2-A3-A4
C      B0=X(12)
C      B4=4.*( .5*(X(10)+X(14))-2.*(X(11)+X(13))+3.*B0)/3.
C      B2=2.*(X(11)+X(13))-2.*B0-.125*B4
C      B3=4.*(X(14)-2.*X(13)+B0-.5*B2-.875*B4)/3.
C      B1=X(14)-B0-B2-B3-B4
C      Y0=X(17)
C      Y4=4.*( .5*(X(15)+X(19))-2.*(X(16)+X(18))+3.*Y0)/3.
C      Y2=2.*(X(16)+X(18))-2.*Y0-.125*Y4
C      Y3=4.*(X(19)-2.*X(18)+Y0-.5*Y2-.875*Y4)/3.
C      Y1=X(19)-Y0-Y2-Y3-Y4
C      FIND ANGLE OF INTERSECTION OF MENISCUS
C      PHI0=PI/2.
C      20 PHI1=PHI0+(PI*X(3)-(PHI0-.5*SIN(2.*PHI0)))/(1.-COS(2.*PHI0))
C      PABS=ABS(PHI1-PHI0)
C      IF(PABS.LT.1.E-4) GO TO 30
C      PHI0=PHI1
C      GO TO 20
C      CALCULATE VIEW AREAS FOR TWO WALL NODES
C      30 L1=X(1)*SIN(PHI1/4.)
C      L2=L1

```

```

L3=X(1)*SIN(PHI1)
L4=.5*X(1)*SQRT(2.*(1.-COS(PHI1)))
L5=.5*X(1)*SQRT((COS(PHI1/2.)-COS(PHI1))**2+(SIN(PHI1/2.)
A +SIN(PHI1))**2)
L6=L4
F35=(L5+L3-L2)/(2.*L3)
F23=L3*(1.-F35)/L2
F45=(L5+L4-L1)/(2.*L4)
F14=L4*(1.-F45)/L1
F26=(L6+L2-L1)/(2.*L2)
F12=L2*(1.-F26)/L1
F13=1.-F12-F14
C
SAF1=SBC*L1*F13
SAF2=SBC*L2*F23
AL=.5*X(1)*SIN(PHI1/2.)
C
C STEADY STATE SOLUTION
ISS=1
T11=(Q*AL/SAF1+TSAT4)**.25
T21=T11
T1S=T11
T2S=T11
T10=T11
T20=T11
55 TN=(T11+T21-460.-2060.)/1600.
K12=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K12=K12*X(2)/(AL*12.)
F1=Q*AL-K12*(T11-T21)-SAF1*(T11**4-TSAT4)
DF1=-K12-4.*SAF1*T11**3
T1N=T11-F1/DF1
ABS1=ABS(T1N-T11)
IF(ABS1.LT.0.001) GO TO 60
T11=T1N
GO TO 55
60 TN=(T11+T21-460.-2060.)/1600.
K12=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K12=K12*X(2)/(AL*12.)
TN=(T21+ T3-460.-2060.)/1600.
K23=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K23=K23*X(2)/(AL*12.)
F2=Q*AL-K12*(T21-T11)-K23*(T21-T3)-SAF2*(T21**4-TSAT4)
DF2=-K12-K23-4.*SAF2*T21**3
T2N=T21-F2/DF2
ABS2=ABS(T2N-T21)
IF(ABS2.LT.0.001) GO TO 70
T21=T2N
GO TO 60
70 ABS1=ABS(T11-T1S)
ABS2=ABS(T21-T2S)
IF(ABS1.LT..001.AND.ABS2.LT.0.001) GO TO 180
T1S=T11
T2S=T21
GO TO 55
C INTEGRATE NODE TEMPERATURES AT 6 SEC INCREMENTS--TEMPERATURES ARE
C IN DEGREES RANKINE
C BEGIN THE ITERATION
90 ISS=0
TIME=0.
T11=520.
T21=520.
T1S=520.
T2S=520.
100 TIME=TIME+0.1
T10=T11
T20=T21
C CALCULATE THERMAL CONDUCTIVITIES AND SPECIFIC HEATS
120 TN=(T11+T21-460.-2060.)/1600.
K12=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K12=K12*X(2)/(AL*12.)
TN=(2.*T11-460.-2060.)/1600.
MCV1=B0+B1*TN+B2*TN**2+B3*TN**3+B4*TN**4

```

```

MCV1=MCV1*.29*X(2)*AL*144.
F1=T11-T10-(Q*AL-K12*(T11-T21)-SAF1*(T11**4-TSAT4))*DT/MCV1
DF1=1.+(K12+4.*SAF1*T11**3)*DT /MCV1
T1N=T11-F1/DF1
ABS1=ABS(T1N-T11)
IF(ABS1.LT.0.001) GO TO 140
T11=T1N
GO TO 120
140 TN=(T11+T21-460.-2060.)/1600.
K12=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K12=K12*X(2)/(AL*12.)
TN=(T21+T3-460.-2060.)/1600.
K23=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
K23=K23*X(2)/(AL*12.)
TN=(2.*T21-460.-2060.)/1600.
MCV2=B0+B1*TN+B2*TN**2+B3*TN**3+B4*TN**4
MCV2=MCV2*.29*X(2)*AL*144.
F2=T21-T20-(Q*AL-K12*(T21-T11)-K23*(T21-T3)-SAF2*(T21**4-TSAT4))*
A *DT/MCV2
DF2=1.+(K12+K23-4.*SAF2*T21**3)*DT /MCV2
T2N=T21-F2/DF2
ABS2=ABS(T2N-T21)
IF(ABS2.LT.0.001) GO TO 160
T21=T2N
GO TO 140
160 ABS1=ABS(T11-T1S)
ABS2=ABS(T21-T2S)
IF(ABS1.LT..001.AND.ABS2.LT..001) GO TO 180
T1S=T11
T2S=T21
GO TO 120
C CALCULATE STRESSES AND COMPARE WITH TENSILE STRENGTH
C ...NOTE THAT THE TEMPERATURES WE'VE SOLVED FOR ARE THOSE OF THE INSID
C SURFACE OF THE WALL
C ASSUME THAT MAXIMUM TEMPERATURES ARE AT TOP OF TANK
180 TN=(2.*T11-460.-2060.)/1600.
K1=A0+A1*TN+A2*TN**2+A3*TN**3+A4*TN**4
SUI=Y0+Y1*TN+Y2*TN**2+Y3*TN**3+Y4*TN**4
T1M=Q*X(2)/(K1*AL*12.)+T11
TN=(2.*T1M-460.-2060.)/1600.
SUD=Y0+Y1*TN+Y2*TN**2+Y3*TN**3+Y4*TN**4
C STRESSES AT INSIDE AND OUTSIDE
DO2=(X(1)+2.*X(2)/12.)*K1
DI2=X(1)*K1
SC=X(4)*DI2/(DO2-DI2)
SRI=SC*(1.-DO2/DI2)
STI=SC*(1.+DO2/DI2)
SZI=SC
SRO=0.0
STO=2.*SC
SZO=SC
FCO=.707107*SQRT((SRO-STO)**2+(STO-SZO)**2+(SZO-SRO)**2)
FCI=.707107*SQRT((SRI-STI)**2+(STI-SZI)**2+(SZI-SRI)**2)
IF(SUD.LE.FCO.OR.SUI.LE.FCI) GO TO 280
IF(ISS.EQ.0) GO TO 100
X(20)=3.
X(25)=0.
GO TO 301
280 IF(ISS.EQ.1) GO TO 40
IF(SUD.GT.FCO) GO TO 300
X(20)=2.
GO TO 301
300 X(20)=1.
301 X(21)=FCO
X(22)=T1M-460.
X(23)=FCI
X(24)=T11-460.
X(25)=TIME
DO 310 IXD=21,26
310 D(IXD)=X(IXD-1)
IOUT=2
CALL CVERT(D,IOUT)

```

```

D(21)=D(21)+0.0001
RETURN
END
SUBROUTINE SVEIW(THETA,HF,S,DT,VF)

THIS SUBROUTINE IS USED BY THE RADIATION FLUX SUBROUTINE.
NO SPECIAL INPUTS ARE REQUIRED FOR THIS ROUTINE BEYOND THOSE
LISTED IN SUBROUTINE JHHRF

RT=DT/2.
X=S-RT
HT=0.0
HO=0.0
XK=X
XM=0.0
PM=0.0
IF(THETA.LE.0.0) GO TO 41
PF = HT/SIN(THETA) + HF*COS(THETA)/SIN(THETA)
XM = PM*COS(THETA)
XF = XM + HF/SIN(THETA)
41 RT=DT/2.
EX = X+RT
IF(THETA) 42,42,43
43 XC = PM + (X-XM)*COS(THETA)
EXC = XC + RT
D = EXC/RT
HFA = (X-XM)*SIN(THETA)
HFB = (XF-X)*SIN(THETA)
GO TO 44
42 EXC = EX
D = EXC/RT
HF1=HT
HF2 = HF + HT
GO TO 95
44 IF (X-XM) 105,205,305
105 HF1 = -HFA
HF2 = HFB
GO TO 95
205 HF2 = HF
HF1 = 0.0
GO TO 95
305 IF (X-XF) 405,605,705
405 HF1 = HFA
HF2 = HFB
GO TO 95
605 HF2 = HF
HF1 = 0.0
GO TO 95
705 HF1 = HFA
HF2 = -HFB
GO TO 95
95 ELV1 = HF1/RT
ELV2 = HF2/RT
A1 = (D+1.)*2 + ELV1*2
A2 = (D+1.)*2 + ELV2*2
B1 = (D-1.)*2 + ELV1*2
B2 = (D-1.)*2 + ELV2*2
ARGA1 = ELV1/((D*2-1.)*.5)
ARGA2 = ELV2/((D*2-1.)*.5)
ARGB1 = ((A1*(D-1.))/(B1*(D+1.)))*.5
ARGB2 = ((A2*(D-1.))/(B2*(D+1.)))*.5
ARGC = ((D-1.)/(D+1.))*2
F1 = (1./((3.1416*D))*ATAN(ARGA1) + (ELV1/3.1416)*((A1-(2.*D))/
1(D*((A1*B1)*.5))*ATAN(ARGB1) - (1./D)*ATAN(ARGC))
F2 = (1./((3.1416*D))*ATAN(ARGA2) + (ELV2/3.1416)*((A2-(2.*D))/
1(D*((A2*B2)*.5))*ATAN(ARGB2) - (1./D)*ATAN(ARGC))
IF (HO) 580,580,582
580 IF(THETA) 118,118,45
582 IF (THETA) 117,117,581
581 IF (XK - XF) 117,116,119
45 IF (X-XM) 118,116,90
90 IF (X-XF) 117,116,119

```

```

116 VF = F2
    GO TO 96
117 VF = F2 + F1
    GO TO 96
118 VF = F2-F1
    GO TO 96
119 VF = F1 - F2
    96 RETURN
    END
    OVERLAY(5,0)
    PROGRAM OV5

```

OV5 EXECUTES THE FOLLOWING INTER-RELATED GROUP OF VAPOR
DISPERSION RATE MODELS -

RATE MODEL	=	C	INDEX	=	3
		G			7
		J			10
		N			14
		S			19
		W			23

COMMON VARIABLES USED - MODNO

SUBROUTINES REQUIRED - MODC1,MODC2,MODG,MODJ,MODN,MODS,MODW,
TRACE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/309A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 8 JANUARY 1976

COMMON/OVCNT/MODNO,OVLST(29),SGLST(29)
INTEGER OVLST,SGLST

```

C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN BRANCH ON MODEL
C      INDEX NUMBER
C      CALL TRACE(0,5,0)
C-----SELECT MODEL C
C      IF(MODNO.NE.3) GO TO 10
C      CALL MODC1
C      CALL MODC2
C      GO TO 100
C-----SELECT MODEL G
C      10 IF(MODNO.NE.7) GO TO 20
C      CALL MODG
C      GO TO 100
C-----SELECT MODEL J
C      20 IF(MODNO.NE.10) GO TO 30
C      CALL MODJ
C      GO TO 100
C-----SELECT MODEL N
C      30 IF(MODNO.NE.14) GO TO 40
C      CALL MODN
C      GO TO 100
C-----SELECT MODEL S
C      40 IF(MODNO.NE.19) GO TO 50
C      CALL MODS
C      GO TO 100
C-----SELECT MODEL W
C      50 IF(MODNO.NE.23) GO TO 100
C      CALL MODW
C

```

```

C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN RETURN TO MAIN
C   HACS CONTROL
C 100 CALL TRACE(1,5,0)
C
C   END
C   SUBROUTINE CONVAP(X,Y,Z,T,H,MDOT,UWIND,IATH,IDIM,SIZE,CHNLW,C)
C *****
C THE NAME OF THIS PROGRAM IS CONVAP. IT RETURNS THE VALUE OF
C CONCENTRATION (GM/CM**3) AT POINT X,Y,Z, AT TIME T FOR VAPOR
C RELEASE FROM A CONTINUOUS SOURCE. BOTH CIRCULAR AND ONE-DIMENSIONAL
C SOURCES ARE CONSIDERED.
C
C INPUT ARGUMENTS
C
C X = WIND DIRECTION CO-ORDINATE CMS
C Y = CROSS WIND CO-ORDINATE CMS
C Z = HEIGHT CO-ORDINATE CMS
C T = TIME AFTER THE SPILL/LEAK AT WHICH THE
C CONCENTRATION IS DESIRED. SECS
C H = HEIGHT OF THE CENTER LINE OF THE SOURCE ABOVE
C GROUND CMS
C MDOT = AVERAGE RATE OF GAS OR VAPOR DISCHARGE GM/SEC
C UWIND = WIND VELOCITY CMS/SEC
C IATH = ATMOSPHERIC CONDITION AS DEFINED IN
C SUBROUTINE JHHDC
C IDIM = DIMENSION OF SPILL (1 IF ONE DIMENSIONAL,
C 2 IF RADIAL)
C SIZE = MAXIMUM RADIUS/LENGTH OF POOL. CMS
C CHNLW = CHANNEL WIDTH (REQUIRED FOR IDIM=1 ONLY) CMS
C
C OUTPUT ARGUMENTS
C
C C = VAPOR CONCENTRATION GMS/CM**3
C *****
C
C REAL MDOT
C PI=3.141592654
C
C CHECKING TO SEE IF THE CLOUD HAS REACHED POINT XYZ OR WHETHER
C THE TRAILING EDGE OF THE CLOUD HAS PASSED THE POINT.
C TRVLT IS THE TRAVEL TIME FOR WIND BETWEEN THE SOURCE AND POINT XYZ.
C
C TRVLT=X/UWIND
C IF(T-TRVLT) 10,20,20
C 10 C=0.0
C RETURN
C 20 VIRDIS=10.*SIZE
C IF(IDIM-1) 40,40,50
C 40 A=CHNLW*SIZE
C VIRDIS=10.*SQRT(A/PI)
C 50 XC=X+VIRDIS
C CALL JHHDC(XC,IATH,SIGY,SIGZ)
C CO=MDOT/(2.*PI*SIGY*SIGZ*UWIND)
C D1=EXP(-1.*((Z-H)*(Z-H)/(2.*SIGZ*SIGZ)))
C D2=EXP(-1.*((Z+H)*(Z+H)/(2.*SIGZ*SIGZ)))
C C3=EXP(-1.*Y*Y/(2.*SIGY*SIGY))
C C=CO*C3*(D1+D2)
C RETURN
C END
C SUBROUTINE ITOX(CHAZ,AM,IFLAG,C)
C
C THIS ROUTINE CONVERTS VAPOR CONCENTRATIONS EXPRESSED IN UNITS
C OF MOLE PERCENT OR PPM TO UNITS OF GM/CM**3.
C *****INPUTS
C CHAZ THE CONCENTRATION OF THE CHEMICAL VAPOR OR GAS IN AIR
C (MOLE PERCENT IF IFLAG IS 0, PPM IF IFLAG IS 1)
C AM MOLECULAR WEIGHT OF THE CHEMICAL
C IFLAG FLAG INDICATING WHETHER MOLE PERCENT OR PPM CONCENTRATION

```

```

C
C      IS BEING GIVEN AS INPUT
C
C****OUTPUTS
C
C      THE CORRESPONDING CONCENTRATION IN GM/CM**3.
10  IF(IFLAG-0) 20,20,10
    CHAZ=CHAZ/1000000.0
    GO TO 25
20  CHAZ=CHAZ/100.0
25  DENV=AM/22414.
    DENA= 0.0012894
    C=1./((1./DENV)+(((1.-CHAZ)*28.9)/(CHAZ*AM*DENA)))
    IF(IFLAG.EQ.0) CHAZ=CHAZ*100.0
    IF(IFLAG.EQ.1) CHAZ=CHAZ*1000000.0
    RETURN
END
SUBROUTINE IVAPC(TMG,UWIND,H,C,X,Y,IAC,IDIM,SW,SIZE,CHNLW,TIME,DUR
1N,JFLAG)
C*****
C      THIS ROUTINE IS THE INVERSION OF ROUTINE VAPC. IT GIVES THE WIDTH
C      OF A GIVEN CONCENTRATION LEVEL IN A VAPOR CLOUD AT ANY POSITION X
C      DOWNWIND AT THE GROUNDLEVEL. IT ALSO RETURNS THE TIME OF ARRIVAL
C      OF THE CLOUD AT DOWNWIND CENTERLINE POSITION X AND THE DURATION
C      FOR WHICH THE VAPOR CONCENTRATION REMAINS ABOVE A HAZARDOUS LEVEL
C      AT A SPECIFIED POSITION X,Y.
C*****
C**** INPUT ARGUEMENTS
C
C      TMG      TOTAL WEIGHT OF VAPOR DISCHARGED      GRAMS
C      UWIND    MEAN WIND VELOCITY IN SPILL AREA      CM/SEC
C      H        HEIGHT OF THE CENTERLINE OF THE DISCHARGE CMS
C      C        HAZARDOUS CONCENTRATION WHOSE CONTOUR WIDTH
C      X        IN THE CLOUD IS DESIRED                GM/CM**3
C      Y        DOWNWIND CENTERLINE DISTANCE AT WHICH THE
C      Y        SPECIFIC CONTOUR WIDTH IS DESIRED      CM
C      Y        ANY CROSSWIND POSITION
C      IAC      (MEASURED FROM CENTERLINE DOWNWIND DIRECTION) CM
C      IAC      ATMOSPHERIC CONDITION FLAG
C      SIZE     ( 1 TO 6 FOR CONDITIONS A TO F RESPECTIVELY)
C      SIZE     THE LENGTH OR RADIUS OF THE VAPOR SOURCE
C      CHNLW    DEPENDING ON IDIM DEFINITION           CM
C      CHNLW    WIDTH OF VAPOR SOURCE ( FOR IDIM=1 ONLY) CM
C
C*** OUTPUT ARGUEMENTS
C
C      SW      HALF WIDTH OF THE ZONE IN THE CLOUD THAT
C      TIME    IS AT OR ABOVE CONCENTRATION C AT X      CM
C      TIME    ELAPSED TIME WHEN CLOUD REACHES DOWNWIND
C      DURN    DISTANCE GIVEN BY X                      SEC
C      DURN    TIME SPAN DURING WHICH CONCENTRATION AT
C      JFLAG   POINT X,Y WILL EXCEED C                  SEC
C      JFLAG   FLAG INDICATING WHETHER POINT X,Y IS INSIDE
C      JFLAG   OR OUTSIDE THE HAZARD ZONE. (1=INSIDE, 2=OUTSIDE)
C*****
C*****
C      PI=3.14159256
C      IF(IDIM-1) 30,30,40
30  A=SIZE*CHNLW
    SIZE=SQRT(A/PI)
40  FACT=10.*SIZE
    X=X+FACT
    CALL JHHDC(X,IAC,SIGY,SIGZ)
    SIGX=SIGY
    C1=(TMG/((2.*PI)**1.5*SIGX*SIGY*SIGZ))
    A1=EXP(-H**2/(2.*SIGZ**2))
    A2=EXP(-H**2/(2.*SIGZ**2))
    C3=C1*A2
    C2=C1*A1
    C0=C2+C3

```

```

      X=X-FACT
      SW=0.
      IF(C,GE,C0) GO TO 5
      SW=SIGY*SQRT(2.*ALOG(C0/C))
5     CONTINUE
      IF(SW.LE.0.0) TIME=0.0
      IF(SW.GT.0.0) TIME=(X-SW)/UWIND
      IF(Y-SW)10,20,20
10    DURN=2.*SQRT(SW**2-Y**2)/UWIND
      IF (IDIM-1) 70,70,80
70    SIZE=A/CHNLW
80    JFLAG=1
      RETURN
20    JFLAG=-1
      DURN=0.
      IF(IDIM-1) 50,50,60
50    SIZE=A/CHNLW
60    RETURN
      END
      SUBROUTINE IVAPCN(X,Y,Z,H,SORDRN,MDOT,SIZE,IDIM,CHNLW,UWIND,IATH,
      *C,SW,TIME,DURN)
C*****
C
C     IVAPCN STANDS FOR INVERSION OF CONTINUOUS VAPOR DISPERSION
C     MODEL. IT RETURNS A VALUE FOR THE MAXIMUM WIDTH TO A
C     CONCENTRATION CONTOUR AT ANY DOWNWIND DISTANCE X AND VERTICAL
C     LOCATION Z. THE SOURCE IS A CONTINUOUS SOURCE. THE ROUTINE
C     ALSO RETURNS A VALUE FOR THE FLAG JFLAG DEPENDING ON WHETHER THE
C     POINT Y IS WITHIN THE CLOUD OR NOT.
C
C     INPUT ARGUMENTS
C
C     X      = DOWN WIND POSITION AT WHICH THE MAXIMUM
C              WIDTH OF HAZARD ZONE IS TO BE KNOWN.          CMS
C     Y      = ANY CROSS WIND POSITION.                        CMS
C     Z      = HEIGHT CO-ORDINATE.                            CMS
C     H      = HEIGHT OF THE CENTER LINE OF THE HOLE ABOVE   CMS
C              GROUND.                                         SECS
C     SORDRN= DURATION FOR WHICH THE SOURCE IS ACTIVE.       GM/SEC
C     MDOT   = AVERAGE RATE OF GAS OR VAPOR DISCHARGE        CMS
C     SIZE   = MAXIMUM RADIUS/LENGTH OF POOL.
C     IDIM   = DIMENSION OF SPILL (1 IF ONE DIMENSIONAL,
C              2 IF RADIAL)
C     CHNLW  = CHANNEL WIDTH (REQUIRED FOR IDIM=1 ONLY)       CMS
C     UWIND  = WIND VELOCITY                                   CMS/SEC
C     IATH   = ATMOSPHERIC CONDITION.
C     C      = HAZARDOUS CONCENTRATION-WHOSE CONTOUR WIDTH   GMS/CM**3
C              IS TO BE KNOWN.
C
C     OUTPUT ARGUMENTS
C
C     SW     = HALF WIDTH OF THE HAZARDOUS ZONE.             CMS
C     TIME   = TIME OF ARRIVAL AT X OF THE HAZARDOUS CONC.   SECS
C     DURN   = DURATION FOR WHICH HAZARDOUS CLOUD AT XYZ     SECS
C*****
C
      REAL MDOT
      PI=3.141592654
      A=PI*SIZE*SIZE
      VIRDIS=10.*SIZE
      IF(IDIM-1) 10,10,20
10    A=CHNLW*SIZE
      VIRDIS=10.*SQRT(A/PI)
20    XC=X+VIRDIS
      CALL JHDC(XC,IATH,SIGY,SIGZ)
      C0=MDOT/(2.*PI*SIGY*SIGZ*UWIND)
      D1=EXP(-1.*((Z-H)*(Z-H)/(2.*SIGZ*SIGZ)))
      D2=EXP(-1.*((Z+H)*(Z+H)/(2.*SIGZ*SIGZ)))
      F=C/(C0*(D1+D2))
      SW=0.

```

```

      CMAX=C0*(D1+D2)
      IF(C-CMAX) 30,40,40
30  SW=SQRT(-2.*SIGY*SIGY*ALOG(F))
40  IF(SW.LE.0.0) TIME=0.0
      IF(SW.GT.0.0) TIME=X/UWIND
      DURN=SORDRN+TIME
      IF(TIME.EQ.0.0) DURN=0.0
      RETURN
      END
      SUBROUTINE JHHDC(XXL,IAC,SIGY,SIGZ)
      XL IS THE X COORDINATE OF POINT WHERE CONCENTRATION IS TO BE
      CALCULATED (INPUT IN CM)
      IAC INDICATES WEATHER CONDITIONS AT TIME OF SPILL (INPUT)
      IAC=1 FOR EXTREMELY UNSTABLE CONDITIONS (A)
      IAC=2 FOR MODERATELY UNSTABLE CONDITIONS (B)
      IAC=3 FOR SLIGHTLY UNSTABLE CONDITIONS (C)
      IAC=4 FOR NEUTRAL CONDITIONS (D)
      IAC=5 FOR SLIGHTLY STABLE CONDITIONS (E)
      IAC=6 FOR MODERATELY STABLE CONDITIONS (F)
      SIGY AND SIGZ ARE DISPERSION COEFFICIENTS (OUTPUTS)

      THE FORMS OF THE EQUATIONS USED IN THIS ROUTINE AND THE
      COEFFICIENTS UTILIZED IN THEM WERE DETERMINED BY LEAST
      SQUARE FITTING PROGRAMS TO BEST REPRESENT THE DISPERSION
      COEFFICIENT CURVES GIVEN BY GIFFORD AND PASQUILL.
      THE SIMPLIFIED EQUATIONS USUALLY FOUND IN THE LITERATURE
      WERE FOUND TO BE CONSIDERABLY IN ERROR UNDER MANY CIRCUMSTANCES.

      DIMENSION AO(6),ADONE(6),ATWO(6),S(5,5),ER(6)
      DATA
      * S(1,1) / .90 /, S(2,1) / .913/, S(3,1) / .919/,
      * S(4,1) / .919/, S(5,1) / .919/,
      * S(1,2) /158. /, S(2,2) /104. /, S(3,2) / 69. /,
      * S(4,2) / 51. /, S(5,2) / 34. /,
      * S(1,3) / 2.041/, S(2,3) / 1.786/, S(3,3) / 1.505/,
      * S(4,3) / 1.332/, S(5,3) / 1.146/,
      * S(1,4) / 1.048/, S(2,4) / .916/, S(3,4) / .737/,
      * S(4,4) / .678/, S(5,4) / .650/,
      * S(1,5) / .041/, S(2,5) / .000/, S(3,5) / -.105/,
      * S(4,5) / -.112/, S(5,5) / -.113/
      DATA AO(1) /-1.1840665/,AO(2)/-1.194544/, AO(3)/-1.1446466/,
      * AO(4) /-1.4521136/,AO(5)/-1.793439/, AO(6)/-2.0571470/
      DATA ADONE(1)/.59323084/,ADONE(2)/1.0679426/,ADONE(3)/1.0130453/,
      * ADONE(4)/1.2195024 /,ADONE(5)/1.3995561/,ADONE(6)/1.47180410/
      DATA
      * ATWO(1)/.23058351/,ATWO(2)/.0033788564/,ATWO(3)/-.012671455/,
      * ATWO(4)/-.0800502 /,ATWO(5)/-.11727572 /,ATWO(6)/-.13377112 /
      DATA ER(1)/.098 /,ER(2)/.1726/,ER(3)/.1116/,
      * ER(4)/.1314/,ER(5)/.1187/,ER(6)/.1186/
      XL=XXL
      XL=XL/100.
      IF(XL.LT.1.0) XL=1.0
      IF(IAC-1) 70,70,20
20  IF(IAC-4) 21,10,10
21  IF(XL-1000.) 30,30,10
30  IAC=IAC-1
      SIGY=S(IAC,2)*((XL/1000.)*S(IAC,1))
      SIGZ=S(IAC,3)+(S(IAC,4)*((ALOG(XL/1000.))/2.303))+(S(IAC,5)*((ALOG
      10(XL/100.))/2.303)**2))
      IAC=IAC+1
      IF(XL-10000.) 60,60,31
31  SIGY=SIGY-(((ALOG(XL)/2.303)-4.)*ER(IAC)*SIGY)
      GO TO 60
10  IAC=IAC-1
      SIGY=S(IAC,2)*((XL/1000.)*S(IAC,1))
      IAC=IAC+1
      IF(XL-10000.) 80,80,83
83  SIGY=SIGY-(((ALOG(XL)/2.303)-4.)*ER(IAC)*SIGY)
80  SIGZ=AD(IAC)+(ADONE(IAC)*(ALOG(XL)/2.303))+(ATWO(IAC)*((ALOG(XL)/
      12.303)**2.))
      GO TO 60
70  SIGY=-.22758999+.84761524*(ALOG(XL)/2.303)

```

```

      SIGY=EXP(2.303*SIGY)
      IF(XL-2000.) 71,71,72
72  SIGY=SIGY+((((ALOG(XL)/2.303)-3.30103)/1.65321251)*ER(1)*SIGY)
71  CONTINUE
      IF(XL-1000.) 81,81,80
81  SIGZ=3.338834+(-2.8047118*(ALOG(XL)/2.303))+(.86706924*((ALOG(XL)/
      12.303)**2.))
      IF(XL-100.0) 82,82,60
82  SIGZ=-1.0055+(0.46852*ALOG(XL))
60  CONTINUE
      SIGZ=EXP(2.303*SIGZ)
      SIGY=SIGY*100.
      SIGZ=SIGZ*100.
      XL=100.*XL
      RETURN
      END
      SUBROUTINE MODC1

```

SUBROUTINE MODC1 IS A PART OF THE GENERALIZED VAPOR DISPERSION MODEL C. IT FINDS THE MAXIMUM EXTENTS OF FIRE AND TOXICITY HAZARDS OF A CLOUD OR PLUME USING ROUTINES VAPC AND CONVAP FOR INSTANTANEOUS AND CONTINUOUS RELEASES RESPECTIVELY. FOR AN INSTANTANEOUS RELEASE, IT ALSO GIVES THE CONCENTRATION VS TIME AT A USER SPECIFIED POINT.

```

      COMMON/C/PLTYP,XBX(150)
      INTEGER PLTYP
      DIMENSION AC(20,2),AX(20),AT(20),ASAVC(20),ASAV(20)
      DIMENSION PTITL(6),PTIT(6),XTITL1(6),XTITL2(6),XTITL3(6),
1  XTITL4(6),YTITL1(6),YTITL2(6)
      EQUIVALENCE (XBX(1),AT(1)),(XBX(81),AX(1)),(XBX(21),AC(1,1))
      EQUIVALENCE (XBX(64),IC1PF),(XBX(61),X),(XBX(62),Y)
      EQUIVALENCE (XBX(63),Z),(XBX(65),AM)
      EQUIVALENCE (XBX(101),ASAVC(1)),(XBX(121),ASAV(1))
      DATA MOD/3H C1 /
      ODATA (PTITL(I),I=1,6)/8HCONCENTR,8HATION VS,8H TIME AT,
      18H A FIXED, H POINT -,8H MODEL C/
      ODATA (PTIT(I),I=1,6)/8HMAX GROU,8HND CONC ,8HVS TIME/,
      18HDISTANCE,8H - MODEL,8H C /
      ODATA (XTITL1(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
      18H.....,8H.....,8HMINUTES)/
      ODATA (XTITL2(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
      18H.....,8H.....,8H.(HOURS)/
      ODATA (XTITL3(I),I=1,6)/8HDOWNWIND,8H DISTANC,8HE.....,
      18H.....,8H.....,8H(METERS)/
      ODATA (XTITL4(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
      18H.....,8H.....,8HMINUTES)/
      ODATA (YTITL1(I),I=1,6)/8HCONCENTR,8HATION ,8HAT POINT,
      18H XYZ ,8H(MOLE PE,8HRCENT) /
      ODATA (YTITL2(I),I=1,6)/8HMAXIMUM ,1H ,8HCONCENTR,
      18HATION ,8H(MOLE PE,8HRCENT) /
1  CONTINUE
      IR=0
      LP=6
      ZR=0.
      IS=6
      TR=0.

```

ACCESS DATA ITEMS

```

      CALL BEGPR(MOD)
      CALL IRCL(2061,INC,IS,IR)
      CALL FRCL(1002,AM,IS,IR)
      CALL FRCL(2012,X,IS,IR)
      CALL FRCL(2013,Y,IS,IR)
      CALL FRCL(2014,Z,IS,IR)
      CALL FRCL(2015,H,IS,IR)
      CALL FRCL(2016,UWIND,IS,IR)
      CALL IRCL(2017,IAC,IS,IR)
      CALL IRCL(2018,IDIM,IS,IR)
      CALL FRCL(2019,SIZE,IS,IR)
      IF(IDIM.EQ.1) CALL FRCL(2020,CHNLW,IS,IR)

```

```

CALL FRCL(2032,CTOX,IS,IR)
CALL FRCL(2033,CFIR,IS,IR)
CALL FRCL(2054,AIRTM,IS,IR)
IF(INC.EQ.0) CALL FRCL(4001,TMG,IS,IR)
IF(INC.EQ.1) CALL FRCL(4044,FLOW,IS,IR)
IF(INC.EQ.1) CALL FRCL(4045,SORDRN,IS,IR)
CALL FRCL(4068,AVTEM,IS,IR)
CALL IRCL(3004,IC1PF,IS,IR)
CALL IRCL(3005,IC2PF,IS,IR)
IW=0
IF(INC.EQ.0) GO TO 93
IF(IC1PF.EQ.1.OR.IC1PF.EQ.3) IW=1
IF(IC2PF.LE.1.AND.IW.EQ.1) IW=2
IF(IC2PF.EQ.0.AND.IW.EQ.2) IC2PF=2
IF(IC2PF.EQ.1.AND.IW.EQ.2) IC2PF=3
IF(IW.EQ.0.OR.IW.EQ.2) GO TO 93
IF(IC2PF.GE.2) IW=3
93 CONTINUE
IF(IC1PF - 2) 4,2,2
2 CALL FRCL(2035,XX,IS,IR)
4 CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 1
CALL OUTPR(MOD)

C
C
C FIND RELATIVE DENSITY OF GAS OR VAPOR WITH AIR AT DISCHARGE
DAIR=(0.079987+(-0.000237*AIRTM))*0.01603
CALL COMPG(AM,AVTEM,DGAS)
RATIO=DGAS/DAIR

C
C
C ITERATE TO FIND MAX EXTENTS OF FIRE AND TOXICITY HAZARDS
IF(CFIR.NE.0.0) CALL ITOX(CFIR,AM,0,CFIRE)
IF(CTOX.NE.0.0) CALL ITOX(CTOX,AM,1,CTOXX)
IXX=1
IF(CFIR.EQ.0.0) XMN=0.0
IF(CFIR.EQ.0.0) GO TO 65
CHAZ=CFIRE
ISTP=0
81 IF(INC.EQ.0.AND.TMG.GT.0.)GO TO 8882
IF(INC.NE.0)GO TO 8883
DX=-32816./6.
GO TO 8883
8882 IF(INC.EQ.0) DX=((6355.*ALOG10(TMG))-32816.)/6.
8883 IF(INC.EQ.1) DX=305.0
IF(DX.LT.305.0) DX=305.0
IF(H.LE.1.0) GO TO 80
IF(IDIM.EQ.2) ADDJ=10.*SIZE
IF(IDIM.EQ.1) ADDJ=10.*SQRT(CHNLW*SIZE/3.14159)
XONE=305.+ADDJ
XTWO=XONE
XJH=XONE
CALL JHHDC(XONE,IAC,SIGY,SIGZ)
SSS=H*H/(2.*SIGZ*SIGZ)
AAAA=100000000.0
IF(SSS.GT.87.) CNC1=0.0
IF(SSS.GT.87.) GO TO 51
IF(INC.EQ.0) CNC1=AAAA/(SIGY*SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
IF(INC.EQ.1) CNC1=AAAA/(SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
51 XTWO=((XTWO-ADDJ)*2.0)+ADDJ
CALL JHHDC(XTWO,IAC,SIGY,SIGZ)
SSS=H*H/(2.*SIGZ*SIGZ)
IF(SSS.GT.87.) CNC2=0.0
IF(SSS.GT.87.) GO TO 57
IF(INC.EQ.0) CNC2=AAAA/(SIGY*SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
IF(INC.EQ.1) CNC2=AAAA/(SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
57 CONTINUE
IF(CNC2.LT.CNC1) GO TO 52
CNC1=CNC2
XONE=XTWO
GO TO 51

```

```

52 XJHH=ABS(XJH-XONE)
   IF(XJHH.LT.0.1) GO TO 47
   A=150.
   IM=0
54 XTHRE=XONE+A
   CALL JHHDC(XTHRE,IAC,SIGY,SIGZ)
   SSS=H*H/(2.*SIGZ*SIGZ)
   IF(SSS.GT.87.) CNC3=0.0
   IF(SSS.GT.87.) GO TO 58
   IF(INC.EQ.0) CNC3=AAAA/(SIGY*SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
   IF(INC.EQ.1) CNC3=AAAA/(SIGY*SIGZ*EXP(H*H/(2.*SIGZ*SIGZ)))
58 CONTINUE
   IF(IM.EQ.1) GO TO 53
   IF(CNC3.LT.CNC1.AND.A.EQ.150.) A=-300.00
   IF(CNC3.GE.CNC1.AND.A.EQ.150.) A=300.00
   IF(CNC3.LT.CNC1.AND.A.EQ.100.) A=-100.0
   IF(CNC3.GE.CNC1.AND.A.EQ.100.) A=100.0
   IM=1
   GO TO 54
53 IF(CNC3.LT.CNC1) GO TO 721
   CNC1=CNC3
   XONE=XTHRE
   IM=1
   GO TO 54
721 B=ABS(A)
   IF(B.EQ.100.) GO TO 56
   IM=0
   A=100.0
   GO TO 54
56 IF(IDIM.EQ.2) XMN=XONE-(10.*SIZE)
   IF(IDIM.EQ.1) XMN=XONE-(10.*SQRT(CHNLW*SIZE/3.14159))
47 IF(INC.EQ.0) CCCC=(CNC1/AAAA)*(2.*TMG/((2.*3.14159265)*1.5))
   IF(INC.EQ.1) CCCC=(CNC1/AAAA)*(FLOW/(3.14159265*UWIND))
   IF(XJHH.LT.0.1) GO TO 80
   GO TO 50
80 XMN=DX
50 TYME=XMN/UWIND
   IF(INC.EQ.0) CALL VAPC(XMN,0.0,0.0,TYME,TMG,H,UWIND,IAC,IDIM,SIZE
1,CHNLW,CNCNT)
   IF(INC.EQ.1) CALL CONVAP(XMN,0.0,0.0,TYME,H,FLOW,UWIND,IAC,IDIM,
:SIZE,CHNLW,CNCNT)
   IF(CNCNT-CHAZ) 55,65,45
45 IF(ISTP.EQ.2) GO TO 55
   SLEFT=XMN
C
C
C   USE GOLDEN SECTION SEARCH TO BRACKET MAX HAZARD EXTENT
   ISTP=1
   XMN=XMN*2.
   SRGT=XMN
   GO TO 50
55 IF(ISTP.EQ.0) GO TO 65
C
C
C   USE BISECTION TECHNIQUE TO ZERO IN ON ANSWER
   IF(CNCNT.LE.CHAZ) SRGT=XMN
   IF(CNCNT.GT.CHAZ) SLEFT=XMN
   IF(SRGT-SLEFT-DX) 66,66,75
66 XMN=SRGT
   GO TO 65
75 ISTP=2
70 XMN=((SRGT-SLEFT)/2.)+SLEFT
   GO TO 50
65 IF(IXX.EQ.1) XFIR=XMN
   IF(IXX.EQ.2) XTOX=XMN
   IF(IXX.EQ.2) GO TO 85
   IXX=2
   IF(CTOX.EQ.0.0) XTOX=0.0
   IF(CTOX.EQ.0.0) GO TO 105
   ISTP=0
   CHAZ=CTOXX
   GO TO 81

```

```

C
C
C      PRINTOUT AND SAVE HAZARD EXTENTS
85  IF(H.LE.1.0) GO TO 105
    IF(CCCC.LT.CFIRE) XFIR=305.
    IF(CCCC.LT.CTOXX) XTOX=305.
105  CALL FSV(4010,XFIR,4)
    CALL FSV(4043,XTOX,4)
    CALL FSV(4069,RATIO,4)
    IF(XFIR.EQ.0.0) CALL PAGER(3)
    IF(XFIR.EQ.0.0) WRITE(LP,100)
    IF(XTOX.EQ.0.0) CALL PAGER(3)
    IF(XTOX.EQ.0.0) WRITE(LP,110)
    IF(XFIR.GT.305.0.AND.XTOX.GT.305.0) GO TO 140
    IF(XFIR.EQ.305.0) CALL PAGER(3)
    IF(XFIR.EQ.305.0) WRITE(LP,120)
    IF(XTOX.EQ.305.0) CALL PAGER(3)
    IF(XTOX.EQ.305.0) WRITE(LP,130)
140  CALL ENDPF(MOD)

C
C
C      DETERMINE WHICH PLOTS OR TABLES ARE DESIRED AND PROCEED TO THE
      APPROPRIATE SECTION OF PROGRAM
      IF(IC1PF.EQ.0.AND.IC2PF.EQ.0) GO TO 40
      IF(IC2PF.EQ.2) GO TO 90
      IF(IC2PF.EQ.3) GO TO 90
      IF(IC1PF.EQ.1) GO TO 90
      IF(IC1PF.EQ.3) GO TO 90
      GO TO 86

C
C
C      CALCULATE DATA FOR PLOT AND TABLE OF CONCENTRATION VS TIME
      AT USER SPECIFIED POINT IF SPILL IS INSTANTANEOUS
90  IF(IC2PF.NE.0.AND.IC2PF.NE.1) GO TO 91
    GO TO 92
91  CALL PAGER(0)
    CALL PAGER(3)
    WRITE(6,95)
    CALL PAGER(1)
    WRITE(6,20)
    CALL PAGER(2)
    WRITE(6,25)
92  CONTINUE
    IF(IDIM.EQ.1) SZ=SQRT(SIZE*CHNLW/3.14159)
    IF(IDIM.EQ.2) SZ=SIZE
    XCEN=X+(10.*SZ)
    CALL JHDC(XCEN,IAC,SIGY,SIGZ)
    TMX=(X+(4.5*SIGY))/UWIND
    TMN=(X-(3.5*SIGY))/UWIND
    IF(TMN.LE.0.0) TMN=1.0
    DT=(TMX-TMN)/19.
    DO 10 I=1,20
      AT(I)=FLOAT(I-1)*DT+TMN
      IF(INC.EQ.1) AT(I)=(X/UWIND)+1.0
      IF(INC.EQ.0) CALL VAPC(X,Y,Z,AT(I),TMG,H,UWIND,IAC,IDIM,SIZE,
        *CHNLW,AC(I,1))
      IF(INC.EQ.1) CALL CONVAP(X,Y,Z,AT(I),H,FLOW,UWIND,IAC,IDIM,SIZE,
        *CHNLW,AC(I,1))
      IF(INC.EQ.1) GO TO 11
10  CONTINUE

C
C
C      WRITE TABLE OF CONCENTRATION VS TIME AT USER SPECIFIED POINT.
11  I=1
175  ATM=AT(I)/60.
    ATH=ATM/60.
    IF(AC(I,1).GT.0.0) CALL TOXIC(AC(I,1),1,AM,XCPPM)
    IF(AC(I,1).GT.0.0) CALL TOXIC(AC(I,1),0,AM,XCMOL)
    IF(XCPPM.GT.1000000.) XCPPM=1000000.
    IF(XCMOL.GT.100.) XCMOL=100.
    IF(IC2PF.NE.0.AND.IC2PF.NE.1) GO TO 28
    GO TO 29

```

```

28 CONTINUE
  CALL PAGER(1)
  WRITE(LP,35) ATM,ATH,XCPM,XCMOL
29 CONTINUE
  IF(INC.EQ.1) GO TO 30
  CALL PAGER(7)
  SORTM=(SORDRN+(X/UWIND))/60.
  CALL PAGER(7)
  WRITE(6,26) SORTM
30 CONTINUE
  IF(INC.EQ.1) GO TO 32
  I=I+1
  IF(I.EQ.21) GO TO 32
  GO TO 175
32 CALL PAGER(4)
  WRITE(LP,31)
  CALL FRCL(2012,X,IS,IR)
  CALL FRCL(2013,Y,IS,IR)
  CALL FRCL(2014,Z,IS,IR)
  IF(IW.GE.2) CALL PAGER(4)
  IF(INC.EQ.1.AND.IW.EQ.2) WRITE(LP,94)
  IF(INC.EQ.1.AND.IW.EQ.3) WRITE(LP,96)
86 CONTINUE
  IF(IC1PF.EQ.0) GO TO 40
  IF(IC1PF.EQ.1) GO TO 14

C
C
  CALCULATE DATA FOR PLOT OF MAXIMUM GROUNDLEVEL CONCENTRATION
  IN DOWNWIND CENTERLINE DIRECTION VS TIME AND DISTANCE
  ZZ=0.
  YY=0.
  IF(INC.EQ.1) XMN=SIZE+1.0
  IF(INC.EQ.1.AND.XMN.LT.305.) XMN=305.
  IF(INC.EQ.1) GO TO 88
  AMT=TMG/908000.
  XMN=(2.**ALOG10(AMT))*1905.
  IF(XMN.LT.SIZE) XMN=SIZE
88 DX=ABS((XMX-XMN)/19.)
  DO 12 I=1,20
  AX(I) = (FLOAT(I-1)*DX)+XMN
  TIM=AX(I)/UWIND
  IF(INC.EQ.0) CALL VAPC(AX(I),YY,ZZ,TIM,TMG,H,UWIND,IAC,IDIM,SIZE,
  *CHNLW,AC(I,2))
  IF(INC.EQ.1) CALL CONVAP(AX(I),YY,ZZ,TIM,H,FLOW,UWIND,IAC,IDIM,SIZ
  *E,CHNLW,AC(I,2))
  IF(AC(I,2).GE.C2X)C2X=AC(I,2)
12 CONTINUE
14 CONTINUE

C
C
  WRITE PLOT FILE
  DO 15 I=1,2
  IF(I.EQ.1.AND.INC.EQ.1) GO TO 15
  IF(IC1PF.EQ.2.AND.I.EQ.1) GO TO 15
  IF(IC1PF.EQ.1.AND.I.EQ.2) GO TO 15
  DO 13 II=1,20
  IF(I.EQ.1) ASAV(II)=AT(II)/60.
  IF(I.EQ.2) ASAV(II)=AX(II)/100.
  IF(AC(II,I).GT.1.) AC(II,I)=1.0
  IF(AC(II,I).GT.0.0) CALL TOXIC(AC(II,I),0,AM,ASAVC(II))
  IF(ASAVC(II).GT.100.) ASAVC(II)=100.
13 CONTINUE
  OIF(I.EQ.1) CALL PLTLP(PTITL,ASAV,ASAVC,20,XTITL1,YTITL1,1,60.,
  1 XTITL2)
  DIV=0.6*UWIND
  OIF(I.EQ.2) CALL PLTLP(PTIT,ASAV,ASAVC,20,XTITL3,YTITL2,1,DIV,
  1 XTITL4)
  CALL PAGER(4)
  WRITE(6,16)
15 CONTINUE

C
C-----SET UP OFF-LINE PLOT
  PLTYP=3

```

```

C
40 CONTINUE
99 RETURN
16 FORMAT(/35X,65H***** RESULTS IN UNITS OF MOLE PERCENT MULTIPLIED
  *BY 10,000 *****/35X,40H***** GIVE ANSWERS IN UNITS OF PPM *****)
20 FORMAT(4X,4TIME,9X,4TIME,8X,8HVAP CONC,5X,8HVAP CONC)
25 FORMAT(3X,6H(MINS),7X,6H( HRS),9X,5H(PPM),7X,7H(MOL P)/)
26 FORMAT(/60H THE DISCHARGE IS CONTINUOUS. THE STEADY-STATE CONCENT
  1RATION/ 40H WILL THEREFORE AVERAGE THAT SHOWN UNTIL,F7.1,14H MINUT
  *ES AFTER/21H THE DISCHARGE STOPS./)
31 FORMAT(/45H THE LOCATION COORDINATES FOR THIS TABLE ARE-/)
35 FORMAT(1X,E10.3,3X,E10.3,3X,E10.3,3X,E10.3)
940FORMAT(/1X,63H***** THIS TABLE REPLACES THE PLOT OF CONCENTRATION
  1VS TIME ****/1X,63H***** REQUESTED IN ORDER TO SAVE COMPUTATION TIM
  2S. ****)
950FORMAT (56H TABLE OF CONCENTRATION VS TIME AT USER SPECIFIED POIN
  17//)
960FORMAT(/1X,63H***** SINCE A PLOT OF CONCENTRATION VS TIME WOULD NO
  1T SHOW ****/1X,63H***** ADDITIONAL INFORMATION, ONE IS NOT PRODUCE
  2D. ****)
1000FORMAT(/5X,56H*** THE MAXIMUM EXTENT OF FLAMMABLE VAPOR HAZARD IS
  1ZERO/9X,56HBECAUSE THE LOWER FLAMMABLE LIMIT CONCENTRATION IS ZERO
  2.)
1100FORMAT(/5X,52H*** THE MAXIMUM EXTENT OF TOXIC VAPOR HAZARD IS ZERO
  1/9X,52HBECAUSE THE LOWER TOXIC LIMIT CONCENTRATION IS ZERO.)
1200FORMAT(/5X,62H*** THE MINIMUM ANSWER HACS CAN GIVE IS 305 CM OR 10
  1 FEET WHEN/9X,61HTHE LOWER FLAMMABLE LIMIT CONCENTRATION IS GREATE
  2R THAN ZERO.)
1300FORMAT(/5X,62H*** THE MINIMUM ANSWER HACS CAN GIVE IS 305 CM OR 10
  1 FEET WHEN/9X,57HTHE LOWER TOXIC LIMIT CONCENTRATION IS GREATER TH
  2AN ZERO.)
END
SUBROUTINE MODC2

```

```

C
C
C
C
C
C
C
SUBROUTINE MODC2 OBTAINS THE NECESSARY DATA TO EXECUTE ROUTINE
IVAPC FOR INSTANTANEOUS SPILLS OR IVAPCN FOR CONTINUOUS SPILLS.
THESE ROUTINES, GIVEN HAZARDOUS VAPOR CONCENTRATIONS, CALCULATE
THE ARRIVAL TIME OF A HAZARDOUS CONCENTRATION, ITS DURATION, AND
ITS WIDTH IN THE CLOUD OR PLUME.

```

```

  DIMENSION ASW(20),AT(20),AX(20),ADRN(20),IID(20)
  DATA MOD/4H C2 /

```

```

5 CONTINUE
  IR=0
  IS=6
  LP=6

```

```

C
C
C
OBTAIN NECESSARY DATA ITEMS

```

```

  CALL BEGPR(MOD)
  CALL IRCL(2061,INC,IS,IR)
  CALL FRCL(1002,AM,IS,IR)
  CALL FRCL(2012,X,IS,IR)
  CALL FRCL(2013,Y,IS,IR)
  CALL FRCL(2015,H,IS,IR)
  CALL FRCL(2016,UWIND,IS,IR)
  CALL IRCL(2017,IAC,IS,IR)
  CALL IRCL(2018,IDIM,IS,IR)
  CALL FRCL(2019,SIZE,IS,IR)
  IF(IDIM.EQ.1) CALL FRCL(2020,CHNLW,IS,IR)
  CALL FRCL(2032,CTOX,IS,IR)
  CALL FRCL(2033,CFIR,IS,IR)
  IF(INC.EQ.0) CALL FRCL(4001,TMG,IS,IR)
  IF(INC.EQ.1) CALL FRCL(4044,FLOW,IS,IR)
  IF(INC.EQ.1) CALL FRCL(4045,SORDRN,IS,IR)
  CALL IRCL(3005,IC2PF,IS,IR)
  IF(IC2PF.EQ.0) GO TO 1
  IF(IC2PF.EQ.2) GO TO 1
  CALL FRCL(2035,XX,IS,IR)
1 CALL EPRNT(MOD,IS,IR,IL)
  IF(IL.EQ.1) GO TO 99
  IF(IL.EQ.2) GO TO 5

```

```

C   CALL IVAPC OR IVAPCN AS APPROPRIATE FOR THE HAZARD CONCENTRATIONS.
C
  IFIR=1
  IITOX=0
  IF(CTOX.GT.0.0) IITOX=1
  IF(CFIR.GT.0.0) IFIR=0
  IF(IITOX.EQ.1) CALL ITOX(CTOX,AM,IITOX,C)
  IF(IITOX.EQ.1) CTOX=C
  IF(IFIR.EQ.0) CALL ITOX(CFIR,AM,IFIR,C)
  IF(IFIR.EQ.0) CFIR=C
  DO 10 ICNT=1,2
  IF(ICNT.EQ.1) C=CTOX
  IF(ICNT.EQ.2) C=CFIR
  IF(IITOX.EQ.0.AND.ICNT.EQ.1) GO TO 10
  IF(IFIR.EQ.1.AND.ICNT.EQ.2) GO TO 10
  IF(INC.EQ.0) CALL IVAPC(TNG,UWIND,H,C,X,Y,IAC,IDIM,SW,SIZE,CHNLW,
1 TIME,DURN,I0)
  IF(INC.EQ.1) CALL IVAPCN(X,Y,0.0,H,SORDRN,FLOW,SIZE,IDIM,CHNLW,UWI
1 ND,IAC,C,SW,TIME,DURN)
  IF(TIME.LT.0.0) TIME=0.0
C
C
C   UPDATE DATA BASE
  CALL OUTPR(MOD)
  IF(ICNT.EQ.2) GO TO 9
  CALL PAGER(2)
  WRITE(LP,100)
  CALL PAGER(2)
  WRITE(6,120)
  CALL FSV(4011,SW,4)
  CALL FSV(4012,DURN,4)
  CALL FSV(4013,TIME,4)
  GO TO 10
9  CALL PAGER(2)
  WRITE(LP,110)
  CALL PAGER(2)
  WRITE(LP,120)
  CALL FSV(4065,SW,4)
  CALL FSV(4066,DURN,4)
  CALL FSV(4067,TIME,4)
10 CONTINUE
  IF(IITOX.EQ.0) CALL PAGER(4)
  IF(IITOX.EQ.0) WRITE(LP,150)
  IF(IFIR.EQ.1) CALL PAGER(4)
  IF(IFIR.EQ.1) WRITE(LP,160)
  CALL ENDPR(MOD)
C
C
C   CALCULATE DATA FOR AND WRITE TABLE OF HALF WIDTHS, ARRIVAL TIMES,
  AND HAZARD DURATIONS IF REQUESTED.
  IF(IC2PF.EQ.0) GO TO 99
  IF(IC2PF.EQ.2) GO TO 99
  IF(INC.EQ.1) XMN=10.*12.*2.54
  IF(INC.EQ.1) GO TO 20
  AMT=TNG/908000.
  XMN=(2.**ALOG10(AMT))*1905.
20 DX=(XMX-XMN)/19.
  DX=ABS(DX)
  DO 50 ICNT=1,2
  IW=0
  IF(IITOX.EQ.0.AND.ICNT.EQ.1) IW=1
  IF(IW.EQ.0) GO TO 22
  CALL PAGER(4)
  WRITE(LP,26)
  GO TO 50
22 IF(IFIR.EQ.1.AND.ICNT.EQ.2) IW=2
  IF(IW.NE.2) GO TO 24
  CALL PAGER(4)
  WRITE(LP,28)
  GO TO 99
24 IF(ICNT.EQ.1) C=CTOX
  IF(ICNT.EQ.2) C=CFIR

```

```

CALL PAGER(0)
CALL PAGER(7)
IF(ICNT.EQ.1) WRITE(6,38)
IF(ICNT.EQ.2) WRITE(6,39)
WRITE(6,41)
WRITE(6,42)
DO 30 I=1,20
  AX(I)=XMN+FLOAT(I-1)*DX
  IF(INC.EQ.0) CALL IVAPC(TMG,UWIND,H,C,AX(I),Y,IAC,IDIM,ASW(I),SIZE
1,CHNLW,AT(I),ADRN(I),IIO(I))
  IF(INC.EQ.1) CALL IVAPCN(AX(I),Y,0.0,H,SORDRN,FLOW,SIZE,IDIM,CHNLW
1,UWIND,IAC,C,ASW(I),AT(I),ADRN(I))
  IF(AT(I).LT.0.0) AT(I)=0.0
  DISM=AX(I)/100.
  DISFT=AX(I)/(2.54*12.)
  ATIM=AT(I)/60.
  SWM=ASW(I)/100.
  SWFT=ASW(I)/(2.54*12.)
  DRNMN=ADRN(I)/60.
  CALL PAGER(1)
  WRITE(6,43) DISM,DISFT,ATIM,SWM,SWFT,DRNMN
30 CONTINUE
  YM=Y/100.
  YFT=Y/(2.54*12.)
  CALL PAGER(3)
  WRITE(6,44) YM,YFT
  CALL PAGER(2)
  WRITE(6,45)
  CALL PAGER(2)
  WRITE(6,46)
50 CONTINUE
99 RETURN
260FORMAT(/1X,50H**** TOXIC VAPOR CLOUD HAZARD TABLE NOT GIVEN ****/
11X,50H**** BECAUSE 2032 LOWER TOXIC LIMIT IS ZERO. ****)
280FORMAT(/1X,54H**** FLAMMABLE VAPOR CLOUD HAZARD TABLE NOT GIVEN *
1***1X,54H**** BECAUSE 2033 LOWER FLAMMABLE LIMIT IS ZERO. ****)
38 FORMAT(/20X,41HTOXIC VAPOR CLOUD HAZARD TABLE - MODEL C2//)
39 FORMAT(/18X,45HFLAMMABLE VAPOR CLOUD HAZARD TABLE - MODEL C2//)
41 FORMAT(1X,10HX-DISTANCE,3X,10HX-DISTANCE,3X,10HARRIV TIME,3X,12H1/
12 HAZ ZONE,3X,12H1/2 HAZ ZONE,3X,8HDURATION)
42 FORMAT(2X,8H(METERS),6X,6H(FEET),6X,9H(MINUTES),5X,8H(METERS),8X,6
1H(FEET),6X,9H(MINUTES)/)
43 FORMAT(1X,G10.4,3X,G10.4,3X,G10.4,4X,G10.4,5X,G10.4,4X,G10.4)
44 FORMAT(/15X,19HTHE Y COORDINATE = ,G10.4,5H M = ,G10.4,4H FT.)
45 FORMAT(15X,41HTHE Z COORDINATE IS FIXED AT GROUNDLEVEL//)
460FORMAT (15X,62HAN ARRIVAL TIME, HALF WIDTH, AND DURATION OF 0.0 IN
1DICATES THE/15X,56HHAZARDOUS CONCENTRATION NEVER REACHES THE GIVEN
2LOCATION.)
100 FORMAT(1X,28HFOR THE TOXIC CONCENTRATION./)
110 FORMAT(1X,44HFOR THE LOWER FLAMMABLE LIMIT CONCENTRATION./)
120 FORMAT(1X,28HAT THE USER SPECIFIED POINT./)
1500FORMAT(/5X,52H*** RESULTS FOR THE TOXIC VAPOR HAZARD ARE NOT GIVEN
1/9X,52HBECAUSE THE LOWER TOXIC LIMIT CONCENTRATION IS ZERO./)
1600FORMAT(/5X,56H*** RESULTS FOR THE FLAMMABLE VAPOR HAZARD ARE NOT G
1IVEN/9X,56HBECAUSE THE LOWER FLAMMABLE LIMIT CONCENTRATION IS ZERO
2./)
END
SUBROUTINE MODG

SUBROUTINE MODG CALLS THE VAPOR DISPERSION ROUTINES OF MODEL C
FOR SPILLS OF LIGHTER-THAN-WATER, INSOLUBLE LIQUIDS WHICH HAVE
BOILING POINTS LESS THAN NORMAL AMBIENT TEMPERATURES. IF THE
SPILL IS INSTANTANEOUS, IT FIRST SUMS THE WEIGHT OF GAS WHICH
ESCAPED FROM A TANK WITH THE WEIGHT OF LIQUID WHICH ESCAPED.
THIS SUM IS UTILIZED IN SUBSEQUENT MODEL C CALCULATIONS.

LP=6
CALL PAGER(5)
WRITE(LP,100)

CALL IRCL(2061,ISPT,IS,IR)

```

```

IF(ISPT.EQ.1) GO TO 10
CALL FRCL(4001,TMG,IS,IR)
CALL FRCL(4002,TML,IS,IR)
TMG=TMG+TML
CALL PAGER(2)
WRITE(LP,102)
CALL FSV(4001,TMG,6)
10 CALL MODC1
CALL MODC2
CALL PAGER(2)
WRITE(LP,101)
RETURN
100 FORMAT(///55H MODEL G IS FUNCTIONALLY REPLACED BY MODELS C1 AND C2
1./)
101 FORMAT(18H MODEL G EXECUTED./)
102 FORMAT(72H FOR MODEL G, LET MASS OF GAS BE TOTAL OF LIQUID AND GAS
*MASSES RELEASED./)
END
SUBROUTINE MODJ

```

C
C
C
C
C
C
C

SUBROUTINE MODJ CALLS THE VAPOR DISPERSION ROUTINES OF MODEL C FOR SPILLS OF HEAVIER-THAN-WATER, INSOLUBLE LIQUIDS WHICH HAVE BOILING POINTS LESS THAN NORMAL AMBIENT TEMPERATURES. IF THE SPILL IS INSTANTANEOUS, IT FIRST SUMS THE WEIGHT OF GAS WHICH ESCAPED FROM A TANK WITH THE WEIGHT OF LIQUID WHICH ESCAPED. THIS SUM IS UTILIZED IN SUBSEQUENT MODEL C CALCULATIONS.

```

LP=6
CALL PAGER(5)
WRITE(LP,100)

```

C
C
C
C

OBTAIN FROM THE EXECUTION OF MODEL A TOTAL MASS GAS AND TOTAL MASS LIQUID FOR USE IN MODELS C1 AND C2

```

CALL IRCL(2061,ISPT,IS,IR)
IF(ISPT.EQ.1) GO TO 10
CALL FRCL(4001,TMG,IS,IR)
CALL FRCL(4002,TML,IS,IR)
TMG=TMG+TML
CALL PAGER(2)
WRITE(LP,102)
CALL FSV(4001,TMG,6)
10 CALL MODC1
CALL MODC2
CALL PAGER(2)
WRITE(LP,101)
RETURN
100 FORMAT(///54H MODEL J IS FUNCTIONALLY REPLACED BY MODELS C1 AND C2
1./)
101 FORMAT(18H MODEL J EXECUTED./)
1020FORMAT (73H FOR MODEL J, LET MASS OF GAS BE TOTAL OF LIQUID AND GA
IS MASSES RELEASED./)
END
SUBROUTINE MODN

```

C
C
C
C
C
C
C
C
C
C
C
C
C
C

SUBROUTINE MODN CALLS THE VAPOR DISPERSION ROUTINES OF MODEL C FOR SPILLS OF SOLUBLE LIQUIDS WITH BOILING POINTS LESS THAN NORMAL AMBIENT TEMPERATURES. IT IS USUALLY PRECEDED BY THE EXECUTION OF MODK. IF MODK HAS BEEN ABLE TO COMPUTE THE AMOUNTS OF CHEMICAL WHICH DISSOLVE AND EVOLVE AS VAPOR, MODN DIRECTLY CALLS MODEL C AFTER ASSUMING THE VAPOR SOURCE IS CIRCULAR AND CALCULATING A POOL RADIUS. IF MODK HAS NOT BEEN ABLE TO COMPUTE SPECIFIC AMOUNT OF CHEMICAL WHICH EVOLVES AS VAPOR, MODN SUMS THE WEIGHT OF GAS RELEASED FROM THE TANK (IF ANY) WITH SOME USER SPECIFIED FRACTION OF THE LIQUID WHICH IS RELEASED FROM THE TANK. THIS SUM IS THEN UTILIZED IN SUBSEQUENT MODEL C CALCULATIONS. IF THE RELEASE TAKES PLACE AT A DEPTH GREATER THAN 10 FEET UNDERWATER OR THE DENSITY OF THE CHEMICAL IS GREATER THAN THAT OF WATER, MODN ASSUMES THE POOL RADIUS IS THE VOLUME OF THE CHEMICAL RELEASED RAISED TO THE ONE-THIRD POWER. IF THE RELEASE IS AT A DEPTH LESS THAN 10 FEET

```

C      UNDERWATER AND THE DENSITY OF THE CHEMICAL IS LESS THAN THAT OF
C      WATER, MODN ASSUMES THE POOL RADIUS IS 12 TIMES THE VOLUME OF
C      CHEMICAL RELEASED RAISED TO THE ONE-THIRD POWER.

      DATA MOD/4H N /
1  CONTINUE
      LP=6
      IS=6
      IR=0
      CALL BEGPR(MOD)
      CALL FRCL(1021,DENL,IS,IR)
      CALL FRCL(2021,H,IS,IR)
      CALL FRCL(4003,VOL,IS,IR)
      IF(VOL.EQ.0.0) CALL FRCL(4002,TML,IS,IR)
      IF(VOL.EQ.0.0) VOL=TML/DENL
      CALL IRCL(2084,IFLAG,IS,IR)
      IF(IFLAG.EQ.1) CALL FRCL(4001,TMG,IS,IR)
      IF(IFLAG.EQ.1) CALL FRCL(4002,TML,IS,IR)
      IF(IFLAG.EQ.1) CALL FRCL(2085,PERC,IS,IR)
      CALL EPRNT(MOD,IS,IR,IL)
      IF(IL.EQ.1) GO TO 99
      IF(IL.EQ.2) GO TO 1
      CALL PAGER(2)
      WRITE(LP,10)
      IF(IFLAG.NE.1) GO TO 5
      TMG=TMG+(PERC*TML)
      CALL PAGER(4)
      WRITE(LP,20) PERC
      CALL FSV(4001,TMG,6)
5  CALL PAGER(3)
      WRITE(LP,30)
      CALL ISV(2018,2,4)
      SIZE=VOL**0.333333
      IF(H.LT.304.8.AND.DENL.LT.1.0) SIZE=12.*VOL**0.333333
      CALL FSV(2019,SIZE,2)
      CALL MODC1
      CALL MODC2
      CALL ENDPR(MOD)
99 RETURN
10 FORMAT(/54H MODEL N IS FUNCTIONALLY REPLACED BY MODELS C1 AND C2.)
20 FORMAT(/63H FOR MODEL N, IT IS ASSUMED THAT THE MASS OF GAS EVOLVE
1D IS THE/26H MASS OF GAS EVOLVED PLUS ,F6.3,1X,34HTIMES THE MASS O
IF LIQUID RELEASED./)
30 FORMAT(/81H THE FOLLOWING VAPOR SOURCE PARAMETERS ARE ESTIMATED FO
1R USE BY MODELS C1 AND C2./)
      END
      SUBROUTINE MODS

      SUBROUTINE MODS CALLS THE VAPOR DISPERSION ROUTINES OF MODEL C
      FOR SPILLS OF SOLUBLE LIQUIDS WITH HIGH VAPOR PRESSURES.

      LP=6
      CALL PAGER(2)
      WRITE(LP,10)
      CALL MODC1
      CALL MODC2
10 FORMAT(/54H MODEL S IS FUNCTIONALLY REPLACED BY MODELS C1 AND C2.)
      RETURN
      END
      SUBROUTINE MODW

      SUBROUTINE MODW CALLS THE VAPOR DISPERSION ROUTINES OF MODEL C
      FOR SPILLS OF LIGHTER-THAN-WATER, INSOLUBLE LIQUIDS WHICH ARE
      VOLATILE AT NORMAL AMBIENT TEMPERATURES. IF THE VAPOR RELEASE IS
      ESTIMATED TO BE BEST REPRESENTED AS BEING INSTANTANEOUS, IT
      COMPUTES THE AMOUNT OF LIQUID WHICH HAS EVAPORATED UP TO THE USER
      SPECIFIED TIME OR THE TIME AT WHICH EVAPORATION STOPS AND PROVIDES
      THIS VALUE TO MODEL C. FOR CONTINUOUS RELEASES, THE MODEL C
      ROUTINES USE THE VAPOR EVOLUTION RATES COMPUTED BY MODEL V.
      MODW ALSO TRANSFERS THE POOL SIZE COMPUTED BY MODEL V FROM FIELD
      NUMBER 4027 TO 2019.

```



```

A2=EXP(-(Z+H)**2/(2.*SIGZ**2))
C3=C1*A2
C2=C1*A1
C0=C2+C3
C=C0*EXP(-(X-UMIND*T)**2/(2.*SIGX**2))*EXP(-Y**2/(2.*SIGY**2))
IF(IDIM-1) 30,30,40
30 SIZE=A/CHNLW
40 RETURN
END
OVERLAY(6,0)
PROGRAM MODD

C
C
C
C
MODD IS USED TO CALCULATE THE SIZE OF A POOL AS A FUNCTION OF
TIME AND THE TIME IT WILL TAKE FOR THE POOL TO COMPLETELY
VAPORIZE. IT IS USED FOR LIGHTER-THAN-WATER, INSOLUBLE CHEMICALS
WITH A BOILING POINT LESS THAN THE AMBIENT TEMPERATURE.

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
DIMENSION SIZ(20),TIM(20),ASAV(20)
DIMENSION PTITL(6),XTITL(6),XTITL1(6),YTITL(6)
EQUIVALENCE (XBX(1),TIM(1)),(XBX(41),SIZ(1)),(XBX(21),ASAV(1))
REAL KC,KHIC
ODATA (PTITL(I),I=1,6)/8HPOOL RAD,8HIUS/LENG,8HTH VS TI,
18HME - MOD,8HEL D ,1H /
ODATA (XTITL(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
18H.....,8H.....(,8HSECONDS)/
ODATA (XTITL1(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
18H.....,8H.....(,8HMINUTES)/
ODATA (YTITL(I),I=1,6)/8HPOOL RAD,8HIUS ,8HOR LENGT,
18HH ,8H(METERS),1H /
DATA MOD/4H D /
1 CONTINUE

C
CALL TRACE(0,6,0)
IR=0
IS=6
LP=6

C
C
C
OBTAIN NECESSARY DATA ITEMS

CALL BEGPR(MOD)
CALL IRCL(2060,ISPT,IS,IR)
CALL FRCL(1003,TCRY,IS,IR)
IF(ISPT.EQ.0) CALL FRCL(1005,VISL,IS,IR)
CALL FRCL(1014,HLATL,IS,IR)
CALL FRCL(1021,DENL,IS,IR)
IF(DENL-1.0) 40,30,30
30 CONTINUE
CALL PAGER(6)
WRITE(LP,100)
DENL=0.99
40 CONTINUE
IF(ISPT.EQ.1) CALL FRCL(2008,DIA,IS,IR)
IF(ISPT.EQ.0) CALL FRCL(2020,CHNLW,IS,IR)
IF(ISPT.EQ.0) CALL IRCL(2022,IQ,IS,IR)
CALL FRCL(2023,TINF,IS,IR)
CALL COMPO(TINF,TCRY,Q)
CALL FSV(2024,Q,4)
CALL FRCL(2024,Q,IS,IR)
IF(ISPT.EQ.1) CALL FRCL(2026,TIME,IS,IR)
IF(ISPT.EQ.1) CALL FRCL(2059,HGT,IS,IR)
IF(ISPT.EQ.0) CALL FRCL(4003,VI,IS,IR)
IF(ISPT.EQ.0.AND.VI.EQ.0.0) CALL FRCL(4002,SPANT,IS,IR)
IF(ISPT.EQ.0.AND.VI.EQ.0.0) VI=SPANT/DENL
IF(ISPT.EQ.1) CALL FRCL(4049,FLOW,IS,IR)
IF(ISPT.EQ.1) CALL FRCL(4050,ENDTH,IS,IR)
CALL IRCL(3006,IP,IS,IR)
CALL IRCL(3013,ITAB,IS,IR)
ICNT=0
CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99

```

```

      IF(IL.EQ.2) GO TO 1
      IF(ISPT.EQ.1) GO TO 80
      TIME=10.0
70  ITC=1
      IDIM=2
      CALL CRYSP(IDIM,IQ,Q,VI,DENL,VISL,HLATL,TCRY,TINF,CHNLW,TIME,ITC,
      *TOWC,KC,KHIC,VOL,SIZE,SIZMX,TEVAP)
      IF(SIZMX.EQ.0.0) TIME=TEVAP/2.
      IF(SIZMX.EQ.0.0) ICNT=ICNT+1
      IF(SIZMX.EQ.0.0.AND.ICNT.EQ.1) GO TO 70
      DIAM=2.*SIZMX
      IF(DIAM.LE.CHNLW) GO TO 16
      IDIM=1
      ITC=1
      CALL CRYSP(IDIM,IQ,Q,VI,DENL,VISL,HLATL,TCRY,TINF,CHNLW,TIME,ITC,
      *TOWC,KC,KHIC,VOL,SIZE,SIZMX,TEVAP)
16  DIAM=2.*SIZMX
      IF(IDIM.EQ.1) DIAM=SQRT(SIZMX*CHNLW*4./3.14159)
      SAVSZ=SIZMX
      GO TO 20
80  IDIM=2
      CALL DSPRD(DENL,DIA,HGT,FLOW,Q,HLATL,TIME,SIZE,TMAX,SIZMX)
      DIAM=2.*SIZMX

      UPDATE DATA BASE WITH OUTPUT

20  CONTINUE
      CALL OUTPR(MOD)
      IF(IDIM.NE.2) GO TO 84
      CALL PAGER(1)
      WRITE(LP,50)
      GO TO 86
84  IF(IDIM.NE.1) GO TO 86
      CALL PAGER(1)
      WRITE(LP,60)
86  CONTINUE
      CALL ISV(2018,IDIM,4)
      IF(ISPT.EQ.0) SIZE=SIZMX
      CALL FSV(2019,SIZE,4)
      CALL FRCL(2019,SIZE,IS,IR)
      IF(ISPT.EQ.0) CALL FSV(4016,TEVAP,4)
      CALL FSV(4007,DIAM,4)
      CALL PAGER(3)
      WRITE(LP,81)
      CALL FSV(4068,TCRY,4)
      IND=0
      IF(ISPT.EQ.1) IND=1
      IF(IND.EQ.1) CALL ISV(2061,IND,4)
      IF(IND.EQ.1) CALL FSV(4044,FLOW,4)
      IF(IND.EQ.1) CALL FSV(4045,ENDTM,4)
      IF(ISPT.EQ.0.AND.TEVAP.GE.600.) IND=2
      IF(IND.EQ.2) FLW=VI*DENL/TEVAP
      IF(IND.EQ.2) CALL ISV(2061,1,4)
      IF(IND.EQ.2) CALL FSV(4044,FLW,4)
      IF(IND.EQ.2) CALL FSV(4045,TEVAP,4)
      IF(IND.EQ.0) CALL ISV(2061,0,4)
      IF(IDIM.EQ.2) SIZE=.5*DIAM
      IF(IDIM.EQ.1) SIZE=DIAM*DIAM*3.14159/(4.*CHNLW)
      CALL PAGER(2)
      WRITE(LP,200)
      SIZE=.707*SIZE
      CALL FSV(2019,SIZE,6)
      CALL ENDPR(MOD)
90  IF(IP.EQ.0.AND.ITAB.EQ.0) GO TO 99

      CALCULATE DATA FOR PLOT AND/OR TABLE OF POOL SIZE VS TIME
      IF REQUESTED.

91  IF(ISPT.EQ.0) DT=TEVAP/20.
      IF(ISPT.EQ.1) DT=ENDTM/20.
      IDIM=2
      STM=0.0

```

```

ITC=1
DO 10 I=1,20
TIM(I)=(FLOAT(I-1)*DT)+DT
11 IF(ISPT.EQ.0) CALL CRYSP(IDIM,IQ,Q,VI,DENL,VISL,HLATL,TCRY,TINF,
*CHNLW,TIM(I),ITC,TOWC,KC,KHIC,VOL,SIZ(I),SIZMX,TEVP)
IF(ISPT.EQ.1) CALL DSPRD(DENL,DIA,HGT,FLOW,Q,HLATL,TIM(I),SIZ(I),
*TMX,SIZMX)
D=2.*SIZ(I)
IF(ISPT.EQ.1) GO TO 10
IF(IDIM.EQ.2.AND.D.GT.CHNLW) IDIM=1
IF(STH.EQ.0.0.AND.D.GT.CHNLW) ITC=1
IF(STH.EQ.0.0.AND.D.GT.CHNLW) STH=TIM(I)
IF(STH.EQ.DT.AND.ITC.EQ.1) GO TO 11
10 CONTINUE
IF(ISPT.EQ.1) GO TO 12
DIFFR=ABS(SIZ(20)-SAVSZ)
IF(DIFFR.GT.3.) SIZ(20)=SAVSZ
IF(DIFFR.GT.3.) STH=TEVAP
12 CONTINUE
C
C      WRITE PLOT FILE
C
IF(IP.EQ.0) GO TO 19
DO 15 I=1,20
15 ASAV(I)=SIZ(I)/100.
CALL PLTLP(PTITL,TIM,ASAV,20,XTITL,YTITL,1,60.,XTITL1)
IF(STH.EQ.0.0.OR.ISPT.EQ.1) GO TO 17
CALL PAGER(2)
WRITE(LP,27) STH
17 CONTINUE
C
C-----SET UP OFF-LINE PLOT
C      PLTYP=5
C
C      WRITE TABLE OF SIZE VS TIME IF REQUESTED.
C
IF(ITAB.EQ.0) GO TO 99
19 CONTINUE
CALL PAGER(0)
CALL PAGER(8)
WRITE(LP,21)
WRITE(LP,22)
WRITE(LP,23)
DO 25 I=1,20
TMNS=TIM(I)/60.
SM=SIZ(I)/100.
SFT=SIZ(I)/(2.54*12.)
CALL PAGER(1)
WRITE(LP,24) TIM(I),TMNS,SIZ(I),SM,SFT
25 CONTINUE
IF(ISPT.NE.1) GO TO 92
CALL PAGER(1)
WRITE(LP,26)
92 CONTINUE
IF(STH.EQ.0.0.OR.ISPT.EQ.1) GO TO 99
CALL PAGER(2)
WRITE(LP,27) STH
GO TO 99
21 FORMAT(/21X,27HPPOOL SIZE VS TIME - MODEL D//)
22 FORMAT( 8X,4HTIME,8X,4HTIME,8X,4HSIZE,8X,4HSIZE,8X,4HSIZE)
23 FORMAT( 7X,6H(SECS),6X,6H(MINS),7X,5H(CMS),8X,3H(M),8X,4H(FT))//)
24 FORMAT( 5X,610.4,2X,610.4,2X,610.4,2X,610.4,2X,610.4,2X,610.4)
26 FORMAT(6X,41HNOTE - THE POOL IS ASSUMED TO BE CIRCULAR)
27 FORMAT(/17X,34H*** NOTE - AT APPROXIMATELY TIME =,610.4,10H SECS,
1***/17X,45H*** THE POOL IS CONFINED BY CHANNEL BANKS ***)
50 FORMAT(29H THE SPILL POOL IS CIRCULAR )
60 FORMAT(46H THE SPILL POOL IS CONFINED BY CHANNEL BANKS )
81 FORMAT( 49H IN CASE MODEL G FOLLOWS, VAPOR SOURCE PARAMETERS/24H A
1RE ESTIMATED AS BEING-)
100 FORMAT(/1X,68HWARNING - THE LIQUID DENSITY OF THE SPILLED CHEMICAL
* IS SO CLOSE TO/ 1X,68HWATER THAT IT MAY OR NOT FLOAT. FOR MODEL
*D, IT WILL BE ASSUMED THAT/ 1X,29HTHE DENSITY IS 0.99 GM/CM**3.//)

```

200 FORMAT(/60H DIM SPILL IS SET TO MEAN POOL SIZE IN CASE MODEL 5 FOL

*LWS.)

99 CONTINUE

CALL TRACE(1,6,0)

END

SUBROUTINE COMPO(TINF,TCRY,Q)

THIS SUBROUTINE CALCULATES THE HEAT FLUX BETWEEN WATER AND A LIQUID WHOSE BOILING POINT IS LESS THAN AMBIENT. THE ESTIMATION IS KNOWN TO BE CORRECT ONLY FOR LIQUEFIED NATURAL GAS. THE FLUXES FOR OTHER SUBSTANCES ARE COMPUTED BY ASSUMING THAT THEIR HEAT TRANSFER COEFFICIENTS ARE EQUAL TO THE EXPERIMENTALLY DETERMINED HEAT FLUX FOR LNG DIVIDED BY THE DIFFERENCE IN TEMPERATURE BETWEEN LNG AND WATER AT 20 DEGREES CENTIGRADE.

*** INPUTS

TINF WATER TEMPERATURE,DEGREES C

TCRY BOILING TEMPERATURE OF LIQUID,DEGREES C

*** OUTPUTS

Q HEAT FLUX,CAL/SEC-CM**2.

Q=(2.5/181.)*ABS(TINF-TCRY)

RETURN

END

SUBROUTINE CRIT(I,ITC,GAM,DEL,TOWC,KHIC,KC)

THIS SUBROUTINE CALCULATES THE CRITICAL TIME, CRITICAL VOLUME AND THE CRITICAL SIZE OF SPREAD FOR SUBSTANCES OF THE TYPE ADDRESSED BY MODEL D. CRITICALITY IS DEFINED AS THE INSTANT IN TIME AT WHICH THE GRAVITY INERTIA SPREAD CHANGES OVER TO THE GRAVITY VISCOUS REGIME. FOR MORE DETAILS SEE THE TECHNICAL REPORT WHICH DESCRIBES THE WORKINGS OF MODEL D. THE NOMENCLATURE USED IN THIS ROUTINE FOR THE VARIOUS QUANTITIES IS THE SAME AS THAT USED IN THE TECHNICAL REPORT.

REAL K,KC,KHI,KHIC,KHIMX

IF(ITC-1) 5,1,5

** TOW=5. IS ARBITRARILY SET TO FIRST FIND OUT THE TIME FOR COMPLE

1 TOW=5.

CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)

DTOW=TOWEN/100.

TOW=TOWEN

2 TOW=TOW-DTOW

CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)

BLT=SQRT(TOW)/GAM

E=ETA-BLT

3 IF(E) 3,4,4

E1=E

4 GO TO 2

TOWC=TOW+DTOW*(E+ABS(E1))

5 CALL GRSPD(I,TOWC,DEL,KHIC,KC,ETA,TOWEN,KHIMX)

RETURN

END

SUBROUTINE CRISP(IDIM,IQ,Q,VI,DENL,VISL,HLATL,TCRY,TINF,CHNLW,TIME
1,ITC,TOWC,KC,KHIC,VOL,SIZE,SIZE,TEVAP)

CRISP STANDS FOR SPREAD OF A CRYOGEN. THIS ROUTINE GIVES THE EXTENT OF SPREAD, VOLUME OF LIQUID REMAINING, MAX SIZE, AND THE TIME FOR COMPLETE EVAPORATION FOR THE SPREAD OF A CRYOGEN ON WATER. THE NOMENCLATURE USED FOR THE VARIOUS PARAMETERS IS THE

```

C      SAME AS THAT USED IN THE TECHNICAL REPORT WHICH DESCRIBES HOW
C      MODEL D WAS DEVELOPED.
C      *****INPUT PARAMETERS*****
C      IDIM      =A FLAG INDICATING WHETHER THE POOL SPREADS RADially
C                  OR IS CONFINED BY CHANNEL BANKS. 1 MEANS IT IS CONFINED
C                  BY CHANNEL BANKS. 2 MEANS IT SPREADS RADially.
C      IQ        =FLAG WHICH SPECIFIES WHETHER THE HEAT TRANSFER IS
C                  LIMITED BY ICE FORMATION UNDER THE SPREADING LIQUID
C                  OR WHETHER IT IS CONSTANT. 1 IS FOR CONSTANT HEAT.
C                  2 IS FOR ICE FORMATION.
C      Q         =VALUE OF THE CONSTANT HEAT FLUX IF IQ=1,CAL/CM**2-SEC
C      VI        =INITIAL VOLUME OF SPILLED LIQUID, CM**3
C      DENL      =DENSITY OF LIQUID, G/CM**3
C      VISL      =VISCOSITY OF SPILLED LIQUID, G/CM-SEC
C      HLATL     =LATENT HEAT OF EVAPORATION OF LIQUID, CAL/G
C      TCRY      =TEMPERATURE OF SPILLED LIQUID, DEG C
C      TINF      =TEMPERATURE OF WATER FAR FROM SPILL INTERFACE,DEG C
C      CHNLW     =WIDTH OF THE CHANNEL WHERE SPILL OCCURS, CM
C                  (REQUIRED ONLY IF SPILL POOL IS CONFINED)
C      TIME      =TIME AT WHICH OUTPUT PARAMETERS ARE DESIRED, SEC
C      ITC       =A FLAG INDICATING WHETHER CERTAIN CALCULATIONS
C                  ARE DESIRED. BY SETTING THIS FLAG TO 1 THE FIRST
C                  TIME THROUGH THE ROUTINE, THESE VALUES ARE COMPUTED.
C                  THEREAFTER, THE FLAG SHOULD BE SET TO SOME OTHER
C                  VALUE TO PREVENT THESE CERTAIN PARAMETERS FROM BEING
C                  RECALCULATED. THIS SAVES CPU TIME IN ITERATION
C                  FOR PLOT AND TABLE ARRAY DEVELOPMENT.
C      *****OUTPUT PARAMETERS*****
C      TOWC      = DIMENSIONLESS TIME AT WHICH THE SPREAD CHANGES
C                  REGIMES
C      KC        =DIMENSIONLESS VOLUME AT CRITICAL TIME TOWC
C      KHIC      =DIMENSIONLESS CRITICAL SPREAD EXTENT
C      VOL       =VOLUME OF LIQUID REMAINING AT SPECIFIED TIME, CM**3
C      SIZE      =EXTENT OF SPREAD AT SPECIFIED TIME, CM
C                  (RADIUS OR LENGTH OF CHANNEL COVERED DEPENDING
C                  UPON THE VALUE OF IDIM GIVEN AS INPUT)
C      SIZMX     =MAX POOL SIZE (RADIUS OR LENGTH AS ABOVE), CM
C      TEVAP     =TIME FOR ALL LIQUID TO EVAPORATE, SEC
C      *****
C      REAL K,KC,KHI,KHIC,KHIM,KICE,L,KHIMX
C      DATA CPI,TF/0.502,0.0/
C      DATA GR,PI,KICE,VISW,HLATI,DENI,DENW/980.,3.14159265,0.005,0.01,
C      180.,0.92,1.0/
C      G=GR*(1.-DENL/DENW)
C      GAHL=(VI*G/(VISL/DENL)**2)**0.25
C      GAMW=(VI*G/(VISW/DENW)**2)**0.25
C      L=VI*(1./3.)
C      IF(IDIM-1) 1,1,2
C      L=SQRT(0.5*VI/CHNLW)
C      HFSEF=HLATI+(TINF-TF)+0.5*CPI*(TF-TCRY)
C      ALFAP=SQRT((KICE*(TF-TCRY)*DENI*HFSEF)/(((HLATL*DENL)**2)*SQRT(VI*
C      1G)))
C      ALFA=2.*PI*ALFAP
C      BETA=(PI/2.)*ALFAP
C      CHTM=SQRT(L/G)
C      DEL=Q/(HLATL*DENL*(L/CHTM))
C      *** SPREAD CALCULATIONS BEGIN ****
C      TOW=TIME/CHTM
C      GO TO (3,14),IDIM
C      GO TO (4,11),IQ
C      *** ONE DIMENSION SPREAD WITH CONSTANT HEAT FLUX ***
C      I=1
C      IF(VISL-0.5*VISW) 8,5,5
C      CALL CRIT(I,ITC,GAMW,DEL,TOWC,KHIC,KC)
C      TOWE=TOWC*(1.+1.375*KC/(TOWC*KHIC*DEL))*(8./11.)
C      KHIM=KHIC*(TOWE/TOWC)**0.375

```

```

        IF(TOW-TOWC) 6,6,7
6      CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
7      KHI=KHIC*(TOW/TOWC)**0.375
        K=KC-(TOWC*KHIC*DEL/1.375)*((TOW/TOWC)**1.375-1.)
        GO TO 1000
8      CALL CRIT(I,ITC,GAML,DEL,TOWC,KHIC,KC)
        TOWE=TOWC*(1.+1.2*KC/(TOWC*KHIC*DEL))**(.5/.6.)
        KHIM=KHIC*(TOWE/TOWC)**0.2
        IF(TOW-TOWC) 9,9,10
9      CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
10     KHI=KHIC*(TOW/TOWC)**0.2
        K=KC-(TOWC*KHIC*DEL/1.2)*((TOW/TOWC)**1.2-1.)
        GO TO 1000
C *** ICE FORMATION **** ONE DIMENSIONAL SPREAD ****
11     I=2
        CALL CRIT(I,ITC,GAML,BETA,TOWC,KHIC,KC)
        TOWE=TOWC*(1.+0.7*KC/(BETA*KHIC*SQRT(TOWC)))*((10./7.))
        KHIM=KHIC*(TOWE/TOWC)**0.2
        IF(TOW-TOWC) 12,12,13
12     CALL GRSPD(I,TOW,BETA,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
13     KHI=KHIC*(TOW/TOWC)**0.2
        K=KC-(10.*BETA/7.)*KHIC*SQRT(TOWC)*((TOW/TOWC)**0.7-1.)
        GO TO 1000
C
C ***** RADIAL SPREAD ***** CONSTANT HEAT FLUX *****
14     GO TO (15,22),IQ
15     I=3
        IF(VISL-VISW*0.5) 19,16,16
16     CALL CRIT(I,ITC,GAMW,DEL,TOWC,KHIC,KC)
        TOWE=TOWC*(1.+1.5*KC/(PI*TOWC*DEL*KHIC**2))*((2./3.))
        KHIM=KHIC*(TOWE/TOWC)**0.25
        IF(TOW-TOWC) 17,17,18
17     CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
18     KHI=KHIC*(TOW/TOWC)**0.25
        K=KC-((DEL*PI/1.5)*TOWC*KHIC**2)*((TOW/TOWC)**1.5-1.)
        GO TO 1000
19     CALL CRIT(I,ITC,GAML,DEL,TOWC,KHIC,KC)
        TOWE=TOWC*(1.+1.25*KC/(PI*TOWC*DEL*KHIC**2))*0.8
        KHIM=KHIC*(TOWE/TOWC)**0.125
        IF(TOW-TOWC) 20,20,21
20     CALL GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
21     KHI=KHIC*(TOW/TOWC)**0.125
        K=KC-(DEL*PI/1.25)*TOWC*KHIC**2*((TOW/TOWC)**1.25-1.)
        GO TO 1000
22     I=4
        CALL CRIT(I,ITC,GAML,ALFA,TOWC,KHIC,KC)
        TOWE=TOWC*(1.+0.75*KC/(PI*ALFA*KHIC**2*SQRT(TOWC)))*((4./3.))
        KHIM=KHIC*(TOWE/TOWC)**0.125
        IF(TOW-TOWC) 23,23,24
23     CALL GRSPD(I,TOW,ALFA,KHI,K,ETA,TOWEN,KHIMX)
        GO TO 1000
24     KHI=KHIC*(TOW/TOWC)**0.125
        K=KC-(4.*PI*ALFA*TOWC*KHIC**2/3.)*((TOW/TOWC)**0.75-1.)
C ***** CONVERSION TO DIMENSIONAL UNITS *****
1000   VOL=K*VI
        SIZE=FLOAT(3-IDIM)*L*KHI
        SIZMX=FLOAT(3-IDIM)*L*KHIM
        TEVAP=TOWE*CHTM
        ITC=0
        RETURN
        END
        SUBROUTINE DSPRD(DENL,DIA,HGT,FLOW,Q,HTVAP,TIME,SIZE,THAX,SIZMX)
C
C *****
C
C THIS SUBROUTINE CALCULATES THE POOL SIZE VERSUS TIME FOR A
C CONTINUOUS DISCHARGE OF A LIGHTER-THAN-WATER INSOLUBLE LIQUID

```

WITH BOILING POINT LESS THAN AMBIENT. IT WORKS ONLY FOR RADIAL
SPREADING.

*** INPUTS ***

DENL	DENSITY OF DISCHARGED LIQUID	GM/CM**3
DIA	DIAMETER OF HOLE	CM
HGT	HEIGHT OF HOLE ABOVE WATER	CM
FLOW	MASS RATE OF FLOW OF DISCHARGED SUBSTANCE	GM/SEC
Q	HEAT FLUX BETWEEN WATER AND CHEMICAL POOL	CAL/SEC-CM**2
HTVAP	HEAT OF VAPORIZATION OF DISCHARGED LIQUID	CAL/GM
TIME	ELAPSED TIME AFTER SPILL STARTS AT WHICH POOL SIZE IS DESIRED	SECS

*** OUTPUTS ***

SIZE	RADIUS OF POOL AT SPECIFIED TIME	CM
TMAX	TIME AT WHICH POOL REACHES MAXIMUM RADIUS	SECS
SIZMX	MAXIMUM RADIUS POOL WILL ATTAIN	CM

REAL K,KHIMX
PI=3.141592654
G=980.
DENW=1.0

CALCULATING THE JET ENTRY PARAMETERS, HYDRAULIC PUMP RADIUS
AND THE RADIAL OUTFLOW VELOCITY.

FLW=FLOW/DENL
GRAV=G*(1.-DENL/DENW)
10 RN=DIA/2.
VEL=FLW/((PI/4.)*DIA*DIA)
FDJET=VEL*VEL/(GRAV*(DIA/2.))
U=SQRT((VEL**2.)+(2.*G*HGT))
A=DIA*SQRT(VEL/U)/2.
HB=A/2.
FB=2.*U*U/(GRAV*A)
FA=8.*FB/((SQRT((8.*FB)+1.))-1.))**3.)
HA=HB*(FB/FA)**(1./3.)
UA=U*HB/HA

CALCULATING THE RADIAL SPREAD PARAMETERS.

TCH=A/UA
TAU=TIME/TCH
F=UA*UA/(HA*GRAV)
E1=0.41
E0=0.68

CRYOGENIC SPREADING ANALYSIS
CALCULATION OF PARAMETERS

YDOT=Q/(HTVAP*DENL)
GAMMA=A*YDOT/(2.*UA*HA)
TCH=A/UA
K=E1*F/(1.-2.*E1)**2
C=SQRT(2./(1.-2.*E1))
EPSILN=K*GAMMA*C*C

MAXIMUM SPREAD CALCULATION

KHIMX=SQRT(1./GAMMA)
TOWBMX=1.6/EPSILN
TMAX=K*TCH*TOWBMX
SIZMX=A*KHIMX
IF(TIME,LT.TMAX) GO TO 40
SIZE=SIZMX
RETURN
40 TOWBAR=TIME/(K*TCH)
ETOWB=EPSILN*TOWBAR

```

      SIGMA=TOWBAR*((1./6.)*ETOWB*ETOWB-0.5*ETOWB+1.)
      PSI=C*SQRT(SIGMA*K)
      SIZE=(1.+PSI)*A
      RETURN
      END
      SUBROUTINE GRSPD(I,TOW,DEL,KHI,K,ETA,TOWEN,KHIMX)
C*****
C      THIS SUBROUTINE RETURNS DIMENSIONLESS VALUES FOR THE EXTENT OF
C      VOLUMEN OF LOQUID REMAINING
C      VOLUME OF LIQUID, TIME FOR COMLETE EVAPORATION ETC., FOR THE SPR
C      CRYOGENIC LIQUID ON WATER.
C*****
C      INPUT PARAMETERS*****
C      *** I = A FLAG WHICH INDICATES WHAT KIND OF SPREAD THE LIQUID IS
C      GOING (I =1 FOR ONE DIMENSIONAL SPREAD WITH CONSTANT HEA
C      2 FOR ONE-DIM SPREAD WITH ICE FORMATION, 3 FOR RADIAL SP
C      WITH CONSTANT HEAT FLUX, AND 4 FOR RADIAL SPREAD WITH IC
C      TION.)
C      *** TOW = DIMENSIONALESS TIME AT WHICH THE VARIOUS QUANTITIES ARE
C      *** DEL = A QUANTITY THAT IS RELATED TO THE HEAT FLUX
C*****
C      OUTPUT PARAMETERS*****
C      *** KHI = DIMENSIONLESS EXTENT OF SPREAD
C      *** K = DIMENSIONLESS VOLUME OF THE LIQUID
C      *** ETA = DIMENSIONLESS THICKNESS OF THE LIQUID FILM DURING SPREAD
C      *** TOWEN= DIMENSIONLESS TIME FOR COMLETE EVAPORATION IF THE SPREA
C      CONTINUES IN THE GRAVITY INERTIA REGIME ONLY.
C      *** KHIMX= MAXIMUM EXTENT OF SPREAD(DIMENSIONLESS) IF THE SPREAD CO
C      IN THE GRAVITY-INERTIA REGIME ONLY.
C*****
C      REAL K,KHI,KHIMX
C      PI=3.14159265
C      GO TO (10,20,30,40),I
C      *** ONE DIMENSIONAL SPREAD WITH CONSTANT HEAT FLUX ***
10    KHI=1.39*TOW**(2./3.)+0.0966*DEL*TOW**(7./3.)
      K=1.-0.834*TOW**(5./3.)*DEL-0.029*(TOW**(10./3.))*DEL**2
      ETA=K/KHI
      KHIMX=1.5874/(DEL**.4)
      TOWEN=1.0891/(DEL**.6)
      RETURN
C      *** ONE DIMENSIONAL SPREAD WITH ICE FORMATION ***
20    KHI=1.39*TOW**(2./3.)
      K=1.-1.19*DEL*TOW**(7./6.)
      ETA=K/KHI
      TOWEN=0.859/(DEL**(6./7.))
      KHIMX=1.39*(TOWEN**(2./3.))
      RETURN
C      *** RADIAL SPREAD WITH CONSTANT HEAT TRANSFER RATE ***
30    KHI=SQRT(1.3*TOW+0.442*DEL*TOW**3)
      K=1.-2.04*DEL*TOW**2-0.347*(DEL**2)*(TOW**4)
      ETA=K/(PI*KHI**2)
      TOWEN=0.6743/SQRT(DEL)
      KHIMX=1.0059/(DEL**0.25)
      RETURN
C      *** RADIAL SPREAD WITH ICE FORMATION ***
40    KHI=SQRT(1.415*DEL*TOW**2.5+1.3*TOW)
      K=1.-0.867*DEL*TOW**1.5-0.4716*(DEL**2)*(TOW**3)
      ETA=K/(PI*KHI**2)
      TOWEN=0.864/(DEL**(2./3.))
      KHIMX=1.451/(DEL**(1./3.))
      RETURN
      END
      OVERLAY(7,0)
      PROGRAM OV7
C
C      OV7 EXECUTES THE FOLLOWING GROUP OF RATE MODELS -
C
C      RATE MODEL = F      INDEX = 6
C                  M      13

```



```

        CALL PAGER(2)
        WRITE(6,100)
        RETURN
100 FORMAT(57H MODEL F HAS BEEN FUNCTIONALLY INCORPORATED INTO MODEL D
1./)
END
SUBROUTINE MODII
C
    CALL PAGER(2)
    WRITE(6,100)
    RETURN
100 FORMAT(45H MODEL FOR INSOLUBLE SOLIDS IS NOT AVAILABLE./)
END
SUBROUTINE MODM
DATA MOD/4H M /
1 CONTINUE
LP=6
IR=0
IS=6
CALL BEGPR(MOD)
CALL IRCL(2084,IFLAG,IS,IR)
CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 1
CALL OUTPR(MOD)
CALL PAGER(4)
WRITE(LP,100)
CALL ISV(2061,0,2)
IF(IFLAG.EQ.1) GO TO 20
CALL PAGER(4)
WRITE(LP,200)
20 CALL ENDPR(MOD)
99 RETURN
100 FORMAT(66H MODEL FOR EVAPORATION RATE OF SOLUBLE LIQUIDS WHOSE BOI
1LING POINT/39H IS LESS THAN AMBIENT IS NOT AVAILABLE./51H IN ORDER
2 TO ASSESS THE VAPOR HAZARD WE ASSUME THAT/40H ALL THE GAS IS EVOL
3VED INSTANTANEOUSLY./)
200 FORMAT(/56H MODEL K HAS ALREADY ESTIMATED THE AMOUNT OF VAPOR WHIC
1H/42H EVOLVES. THIS AMOUNT IS USED IN MODEL N./)
END
SUBROUTINE MODO
CALL PAGER(7)
WRITE(6,100)
RETURN
100 FORMAT(58H MODEL FOR HEAT RELEASE FROM LIQUIDS THAT REACT WITH WAT
1ER/18H IS NOT AVAILABLE./55H CONSULT MANUAL 2 TO DETERMINE THE PRO
2DUCTS OF REACTION/27H OF THE CHEMICAL AND WATER./62H HAZARDS MAY B
3E ESTIMATED SEPARATELY IF THE PRODUCTS HAVE BEEN/33H INCLUDED ON T
4HE PROPERTIES FILE./)
END
SUBROUTINE MODRR
C
    CALL PAGER(2)
    WRITE(6,100)
    RETURN
100 FORMAT(44H MODEL FOR REACTIVE SOLIDS IS NOT AVAILABLE./)
END
SUBROUTINE MODSS
C
    CALL PAGER(2)
    WRITE(6,100)
    RETURN
100 FORMAT(45H MODEL FOR INSOLUBLE SOLIDS IS NOT AVAILABLE./)
END
SUBROUTINE MODY
C
    CALL PAGER(2)
    WRITE(6,100)
    RETURN
100 FORMAT(24H MODEL Y DOES NOT EXIST./)
END
SUBROUTINE MODZ

```

```

C
  CALL PAGER(2)
  WRITE(6,100)
  RETURN
100 FORMAT(53H MODEL FOR SELF REACTING SUBSTANCES IS NOT AVAILABLE./)
  END
  OVERLAY(10,0)
  PROGRAM OVB

C
  PROGRAM OVB EXECUTES THE FOLLOWING GROUP OF RATE MODELS -

      RATE MODEL =      INDEX =
          I          9
          K          11
          P          16
          R          18
          T          20
          V          22
          X          24

C
  COMMON VARIABLES USED - MODNO
  SUBROUTINES REQUIRED - MODK,MODP,MODR,MODT,MODV,TRACE .
  AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
          35/309A ACORN PARK,
          CAMBRIDGE, MASS., 02140
          TEL. 617-864-5770 EXT. 2813
  DATE - 8 JANUARY 1976

C
  COMMON/OVER/NOV,SEG
  COMMON/OVCNT/MODNO,OVLST(29),SGLST(29)
  INTEGER      OVLST,SGLST

C
C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN BRANCH ON MODEL
C      INDEX NUMBER
C      CALL TRACE(0,8,0)

C
C-----SELECT MODEL I
C      IF(MODNO.NE.9) GO TO 10
C      SEG=1
C      CALL SEGLOD(1)
C      GO TO 100

C
C-----SELECT MODEL K
C      10 IF(MODNO.NE.11) GO TO 20
C      SEG=2
C      CALL SEGLOD(2)
C      GO TO 100

C
C-----SELECT MODEL P
C      20 IF(MODNO.NE.16) GO TO 30
C      SEG=2
C      CALL SEGLOD(2)
C      GO TO 100

C
C-----SELECT MODEL R
C      30 IF(MODNO.NE.18) GO TO 40
C      SEG=3
C      CALL SEGLOD(3)
C      GO TO 100

C
C-----SELECT MODEL T
C      40 IF(MODNO.NE.20) GO TO 50
C      SEG=4
C      CALL SEGLOD(4)
C      GO TO 100

C
C-----SELECT MODEL V
C      50 IF(MODNO.NE.22) GO TO 60

```

```

SEG=5
CALL SEGLOD(5)
GO TO 100
C-----SELECT MODEL X
60 IF(MODNO.NE.24) GO TO 100
SEG=6
CALL SEGLOD(6)
C-----PRINT OVERLAY EXECUTION TRACE MESSAGE, THEN RETURN TO MAIN
HACS CONTROL
100 CALL TRACE(1,8,0)
C
END
SUBROUTINE CDIFW(AM,DENL,TWAT,TCRIT,TBOIL,DIFCO)
*****
THIS SUBROUTINE ESTIMATES THE DIFFUSION COEFFICIENT FOR A LIQUID
CHEMICAL IN WATER
*** INPUTS ***
      AM      MOLECULAR WEIGHT OF CHEMICAL      GM/MOLE
      DENL     DENSITY OF CHEMICAL AT ITS BOILING POINT  GM/CM3
      TWAT     TEMPERATURE OF WATER              DEG. C
      TCRIT    CRITICAL TEMPERATURE OF CHEMICAL      DEG. C
      TBOIL    BOILING TEMPERATURE OF CHEMICAL      DEG. C
*** OUTPUTS ***
      DIFCO    DIFFUSION COEFFICIENT OF CHEMICAL IN WATER  CM2/SEC
*****
      TR=(TWAT+273.2)/(TCRIT+273.2)
      TBR=(TBOIL+273.2)/(TCRIT+273.2)
      MOLAL VOLUME (VB) CALCULATION IS MATHIAS METHOD FOUND ON PG 106
      OF 2ND ED. OF REID AND SHERWOOD, THE PROPS OF GASES AND LIQUIDS
      VB=(AM/DENL)*((2.-TR)/(2.-TBR))
      VISCOSITY OF WATER EQUATION IS FROM PAGE 374 OF THE THIRD EDITION
      OF PERRY'S CHEMICAL ENGINEERING HANDBOOK
      VISW=(2.1482*((TWAT-8.435)+SQRT(8078.4+((TWAT-8.435)**2.)))-120.
      VISW=100./VISW
      DIFFUSION COEFFICIENT EQUATION IS WILKE AND CHANG METHOD FOUND
      IN REID AND SHERWOOD ON PG 549.
      DIFCO=5.06E-07*(TWAT+273.2)/((VISW**1.1)*(VB**0.6))
      RETURN
END
SUBROUTINE COMPD(AM,TA,DENLB,DIFCO)
THIS SUBROUTINE CALCULATES THE DIFFUSION COEFFICIENT OF A VAPOR IN
AND SHOULD BE THE DEFAULT VALUE IF THE DATA IS NOT PRESENT IN THE
MANUAL 2 DATA FILE.
***INPUTS
      AM      THE MOLECULAR WEIGHT OF THE CHEMICAL
      TA      AMBIENT TEMPERATURE,DEGREES C
      DENLB   THE DENSITY OF THE LIQUID AT ITS BOILING POINT,GM/CM**3.
***OUTPUTS
      DIFCO   THE DIFFUSION COEFFICIENT,CM**2./SEC
      VB=29.9**33333
      WMO=29.0
      VA=AM/DENLB

```

```

T=(TA+273.2)**1.5
S1=((1./WMO)+(1./AM))**.50
S2=((VA**.33333)+VB)**2.
DIFCO=.0043*(T/S2)*S1
RETURN
END
SUBROUTINE CRITD(AVP,BVP,CVP,TEMP,DEPTH)
C *****
C THIS SUBROUTINE CALCULATES THE DEPTH IN WATER AT WHICH A LIQUID
C WITH A BOILING POINT LESS-THAN-AMBIENT TEMPERATURE WILL NOT BOIL
C BECAUSE OF THE HYDROSTATIC PRESSURE ACTING UPON IT. IT IS
C PRIMARILY USEFUL FOR HEAVIER-THAN-WATER CHEMICALS SINCE LIGHTER
C SUBSTANCES WILL RISE UNTIL THEY REACH A DEPTH AT WHICH THEY CAN
C BOIL.
C *** INPUTS ***
C AVP      COEFFICIENT OF VAPOR PRESSURE EQUATION WHICH
C          GIVES ANSWER IN MM HG.
C BVP      - - - - -
C CVP      - - - - -
C TEMP     TEMPERATURE OF CHEMICAL                      DEG. C
C *** OUTPUTS ***
C DEPTH     DEPTH AT WHICH CHEMICAL WILL NOT BOIL          CMS
C *****
C GRAV=980.7
C DENL=1.0
C VAP=1333.224*10.**(AVP-(BVP/(TEMP+CVP)))
C PATH=760.*1333.224
C IF(VAP.LE.PATH) DEPTH=0.0
C IF(VAP.LE.PATH) GO TO 99
C DEPTH=(VAP-PATH)/(DENL*GRAV)
99 RETURN
END
SUBROUTINE DISP(W,D,IFLAG,T,UF,UT,XN,TP,E,EX,EY,EZ)
C *****
C *** THIS SUBROUTINE IS CALLED BY THE DILUN SUBROUTINE , DISPERSION
C AND TURBULENT DIFFUSION COEFFICIENTS ARE RETURNED BY THIS SUBROU
C
C PI=3.14159265
C B=W/2.
C IF(IFLAG.EQ.1) GO TO 60
C RH=W*D/(2.*D+W)
C GO TO (60,70,80),IFLAG
60 E=0.
C RETURN
C *** SPILL INTO A NON TIDAL RIVER
70 USTAR=6.7305*XN*UF/RH**(1./6.)
C EZ=0.067*USTAR*RH
C EX=0.1*EZ
C IF(W/D-100.) 72,71,71
71 EY=0.1*EZ
C E=136.09*XN*UF*RH**(5./6.)
C GO TO 75
72 EY=0.23*USTAR*RH
C E=225.*USTAR*RH
75 RETURN
C *** TIDAL RIVER *****
80 USTAR=6.7305*XN*(2.*UT/PI)/RH**(1./6.)
C *** USTAR IS BASED ON THE MEAN OSCILLATING FLOW VELOCITY. ***
C EZ=0.067*USTAR*RH
C EX=0.1*EZ
C EY=0.1*EZ
C IF(W/D-100.) 81,82,82
81 EY=0.23*USTAR*RH
C ** TRANSVERSE AND VERTICAL DISPERSION COEFFICIENTS
82 EV=6.*D*USTAR

```

```

      ET=0.011*0.025*EV*(UT*TP/B)**2
      TPV=TP/(D**2/EZ)
      TPT=TP/(B**2/EY)
      IF(TPV-1.) 83,83,84
83      GO TO 85
84      IF(ET/EV-1.) 85,85,86
85      E=EV
      GO TO 87
86      E=ET
87      RETURN
      END
      FUNCTION ERF(X)
      DIMENSION A(5)
      DATA P/0.3275911/,
1      A(1)/0.254829592/,A(2)/-0.284496736/,A(3)/1.421413741/,
2      A(4)/-1.453152027/,A(5)/1.061405429/
      .....
      ERF COMPUTES THE ERROR FUNCTION OF X BY SERIES EXPANSION WITH AN
      ERROR LESS THAN OR EQUAL TO 1.5*10-7.
      REFERENCE.. C. HASTINGS JR.
                  APPROXIMATIONS FOR DIGITAL COMPUTERS
                  PRINCETON UNIVERSITY PRESS
                  PRINCETON N.J. 1955
      .....
      CODE REVISIONS INSERTED 3 NOV 1978 BY R.G. POTTS
      INCLUDES LIMIT TEST FOR ASYMPTOTIC VALUE
      ONE=1.0
      IF(X.LT.0.0) ONE=-ONE
      T=ONE/(ONE+P*X)
      EX=EXP(-X*X)
      ERF=1.0-(T*(A(1)+T*(A(2)+T*(A(3)+T*(A(4)+T*(A(5)))))*EX)
      IF(ABS(1.0-ERF).LT.1.5E-7) ERF=1.0
      ERF=ONE*ERF
      RETURN
      END
      SUBROUTINE HNTC(DIFCO,XMOL,VOLI,HMP)
      ANU=0.15
      AL=VOL**0.3333
      CUT=5.*10.**5
      TEMP= 293.
      RVAP=82.057/XMOL
      SCHM=ANU/DIFCO
      VELOC= 450.
      REYN=(VELOC*AL)/ANU
      IF(REYN-CUT) 1,1,2
1      HBAR=(1.328*(REYN**.5)*DIFCO*(SCHM**.3333))/AL
      GO TO 3
2      HBAR=(.037*DIFCO*(SCHM**.3333)*(REYN**.8))/AL
3      HMP=HBAR/(RVAP*TEMP)
      RETURN
      END
      SUBROUTINE RLJSP(IDIM,VOL,DENL,VISL,SURT,TIME,CHNLW,SIZE)
      THIS SUBROUTINE CALCULATES THE SPREAD OF LIQUIDS ON WATER
      *** INPUTS
      IDIM  DIMENSION OF SPILL (1 IF ONE DIMENSIONAL, 2 IF RADIAL)
      VOL   QUANTITY OF SPILL CM**3
      DENL  DENSITY OF LIQUID GM/CM**3
      VISL  VISCOSITY OF LIQUID GM/CM-SEC
      SURT  SURFACE TENSION WITH SEA WATER DYNES/CM
      TIME  TIME SEC
      CHNLW CHANNEL WIDTH (REQUIRED FOR IDIM=1 ONLY) CM
      *** OUTPUTS
      SIZE  SIZE OF SPILL AFTER TIME HAS ELAPSED (RADIUS OR LENGTH)-CM

```

```

DATA GR,DENW,VISW/980.,1.,.01/
C....CALCULATE NON-DIMENSIONAL PARAMETERS
GO TO (5,10),IDIM
5  AL=SQRT(.5*VOL/CHNLW)
GO TO 15
10 AL=VOL**.333333
15 G=GR*ABS(DENL-DENW)/DENW
T=TIME/SQRT(AL/G)
IF(VISW*.2 - VISL) 25,25,85
C....START CALCULATIONS FOR THE CASE WHEN THE LIQUID VISCOSITY IS
C....GREATER THAN THE VISCOSITY OF WATER
25 A=(AL**3*G)**.25/SQRT(VISW/DENW)
B=SURT/(VISW*SQRT(AL*G))
GO TO (35,60),IDIM
C....FOR THE ONE DIMENSIONAL CASE
35 IF(T-A**(6./7.))40,40,45
40 S=1.39*T**.666666
GO TO 140
45 IF(T-(.972*A**.75/SQRT(B))**.2.66666)50,50,55
50 S=1.39*A**.25*T**.375
GO TO 140
55 S=1.43*SQRT(B/A)*T**.75
GO TO 140
C....FOR RADIAL SPREADING
60 IF(T-(.86*A**.166666)**4)65,65,70
65 S=1.14*SQRT(T)
GO TO 140
70 IF(T-(.61*A**.666666)**2/B)75,75,80
75 S=.98*A**.166666*T**.25
GO TO 140
80 S=1.6*SQRT(B/A)*T**.75
GO TO 140
C....START CALCULATIONS FOR THE CASE WHEN THE LIQUID VISCOSITY IS LESS
C....THAN THE VISCOSITY OF WATER
85 A=(AL**3*G)**.25/SQRT(VISL/DENL)
B=SURT/(VISL*SQRT(AL*G))
GO TO (90,115),IDIM
C....FOR THE ONE DIMENSIONAL CASE
90 IF(T-(.81*A**.4)**(15./7.))40,40,100
100 IF(T-(.90*A**.4/B**.333333)**7.5)105,105,110
105 S=1.13*A**.4*T**.2
GO TO 140
110 S=1.26*(B*T)**.333333
GO TO 140
C....FOR RADIAL SPREADING
115 IF(T-(.685*A**.25)**2.66666)65,65,125
125 IF(T-(.735*(A/B)**.25)**8)130,130,135
130 S=.78*A**.25*T**.125
GO TO 140
135 S=1.062*(B*T)**.25
C....CALCULATE SIZE AND RETURN
140 SIZE=S*AL*FLOAT(3-IDIM)
RETURN
END
SUBROUTINE SOLUB(TEMP,CSAT,IS,IR)
*****
C THIS SUBROUTINE INTERROGATES THE STATE FILE TO DETERMINE THE
C AVAILABILITY OF CHEMICAL SOLUBILITY DATA. IF THE COEFFICIENTS
C FOR THE SOLUBILITY AS A FUNCTION OF TEMPERATURE EQUATION ARE
C PRESENT, IT UTILIZES THEM TO CALCULATE THE SOLUBILITY AT THE
C SPECIFIED TEMPERATURE. IF THEY ARE NOT PRESENT, IT RETURNS THE
C SOLUBILITY AT A FIXED TEMPERATURE WHICH MIGHT ALTERNATIVELY BE
C STORED IN THE DATA BASE.
C
C *** INPUTS ***
C TEMP TEMPERATURE AT WHICH SOLUBILITY DESIRED DEG C
C
C *** OUTPUTS ***
C CSAT SOLUBILITY OF CHEMICAL GM SOLUTE/ 100 GMS SOLVENT

```

```

C      IR      CODE FOR CONFIRMING VALIDITY OF DATA RECALLED FROM
C      SUBROUTINE FRCL
C      IS      SOURCE CODE OF OUTPUT OF FRCL
C
*****
IND=0
ISAVS=IS
ISAVR=IR
IS1=8
IER1=0
CALL FRCL(1028,SOL1,IS1,IER1)
IF(IER1.EQ.1) IR=1
IF(IS1.LT.IS) IS=IS1
IS2=8
IER2=0
CALL FRCL(1029,SOL2,IS2,IER2)
IF(IER2.EQ.1) IR=1
IF(IS2.LT.IS) IS=IS2
IF(IS1.GT.1.AND.IS2.GT.1) CSAT=SOL1+(SOL2*(TEMP+273.16))
IF(IS1.GT.1.AND.IS2.GT.1) RETURN
CALL PAGER(4)
WRITE(6,10)
IF(IER1.EQ.1.OR.IER2.EQ.1) IND=1
IF(IND.EQ.1.AND.ISAVR.EQ.0) IR=0
IF(IS1.LT.ISAVS.OR.IS2.LT.ISAVS) IS=ISAVS
IS3=8
IER3=0
CALL FRCL(1026,CSAT,IS3,IER3)
IF(IER3.EQ.1) IR=1
IF(IS3.LT.IS) IS=IS3
RETURN
10 FORMAT(/1X,62HTHE SOLUBILITY EQUATION COEFFICIENTS ARE NOT IN THE
*DATA BASE./,1X,51HTHE SOLUBILITY AT A FIXED TEMP IS THEREFORE CALL
*ED./)
END
OVERLAY(10,1)
PROGRAM MODI

PROGRAM EXECUTES MODEL I, INDEX 9

MODI OBTAINS THE NECESSARY DATA FOR EXECUTION OF SUBROUTINE
EVDPR, WHICH CALCULATES THE RATE OF EVAPORATION THAT IS
SPILLED ON THE WATER SURFACE

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
DIMENSION AT(20),AV(20),AEV(20),ASAV(20)
DIMENSION PTITL(6),PTIT(6),XTITL(6),XTITL1(6),YTITL(6),YTITL1(6)
EQUIVALENCE (XBX(1),AT(1)),(XBX(41),AEV(1)),(XBX(21),AV(1))
EQUIVALENCE (XBX(61),ASAV(1))
DATA MOD/4H I /
ODATA (PTITL(I),I=1,6)/8HVOLUME 0,8HF LIQUID,8H REMAINI,
18HNG VS TI,8HME - MOD,8HEL I /
ODATA (PTIT(I),I=1,6)/8HEVAPORAT,8HION RATE,8H VS TIME,
18H - MODEL,2H I,1H /
ODATA (XTITL(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
18H.....,8H.....(,8HSECONDS)/
ODATA (XTITL1(I),I=1,6)/8HELAPSED ,8HTIME....,8H.....,
18H.....,8H.....(,8HMINUTES)/
ODATA (YTITL(I),I=1,6)/8HVOLUME ,8H ,8HREMAININ,
18HG ,8H(M**3) ,1H /
ODATA (YTITL1(I),I=1,6)/8HEVAPORAT,8HION ,8HRATE ,
18H ,8H(KG/SEC),1H /
1 CONTINUE

CALL TRACE(0,8,1)
LP=6
IR=0
IS=6

```

```

C      OBTAIN DATA
C
CALL BEGPR(MOD)
CALL FRCL(1003,TBOIL,IS,IR)
CALL FRCL(1008,SURT,IS,IR)
CALL FRCL(1010,AVP,IS,IR)
CALL FRCL(1011,BVP,IS,IR)
CALL FRCL(1012,CVP,IS,IR)
CALL FRCL(1014,XLAT,IS,IR)
CALL FRCL(1021,DENL,IS,IR)
IF(DENL-1.0)20,20,30
20 CALL PAGER(6)
   WRITE(LP,100)
   DENL=1.01
30 CONTINUE
   CALL FRCL(2023,TINF,IS,IR)
   CALL FRCL(2036,TCRY,IS,IR)
   CALL FRCL(4002,TML,IS,IR)
   VI=TML/DENL
   CALL FRCL(4050,TIMEL,IS,IR)
   CALL IRCL(3007,IIPF,IS,IR)
   CALL IRCL(3016,ITAB,IS,IR)
   CALL EPRNT(MOD,IS,IR,IL)
   IF(IL.EQ.1) GO TO 99
   IF(IL.EQ.2) GO TO 1
C
      CALL EVDRP
      TIME=1000000.0
      CALL EVDRP(VI,DENL,SURT,XLAT,TINF,TCRY,TIME,V,TOTEV,TIMCR)
C
      CALL CRITICAL DEPTH CALCULATION ROUTINE
C
      TMX=TINF
      IF(TCRY.GT.TMX) TMX=TCRY
C
      THE GREATER OF THE WATER TEMPERATURE OR THE CARGO TEMPERATURE
      IS USED IN THE CALCULATION
C
      CALL CRITD(AVP,BVP,CVP,TMX,DEPTH)
C
      UPDATE STATE VECTOR
C
      CALL OUTPR(MOD)
      CALL FSV(4021,TIMCR,4)
      CALL FSV(4046,DEPTH,4)
      CALL ENDPR(MOD)
      CALL PAGER(3)
      WRITE(LP,46)
      CALL ISV(2018,2,4)
      CALL FSV(2019,150.,4)
      CALL FSV(4068,TBOIL,4)
      ISP=0
      TIMET=TIMCR+TIMEL
      IF(TIMET.LT.600.) GO TO 36
      FLOW=TML/TIMET
      ISP=1
      CALL FSV(4044,FLOW,4)
      CALL FSV(4045,TIMET,4)
36 CALL ISV(2061,ISP,4)
C
      INTERROGATE USER PLOT AND TABLE FLAGS
      IF(IIPF.EQ.0.AND.ITAB.EQ.0) GO TO 99
      DT=TIMCR/19.
C
      SET UP LOOP TO CALCULATE PLOT ARRAYS OF TIME VERSUS EVAPORATIO
C
      DO 10 I=1,20
      AT(I) = FLOAT(I-1)*DT
      CALL EVDRP(VI,DENL,SURT,XLAT,TINF,TCRY,AT(I),AV(I),AEV(I),TIMCR)
10 CONTINUE
      IF(IIPF.NE.1) GO TO 40

```

```

DO 16 II=1,2
DO 15 I=1,20
IF(II,EQ.1) ASAV(I)=AV(I)/1000000.
15 IF(II,EQ.2) ASAV(I)=AEV(I)/1000.
DIV=60.
IF(II,EQ.1) CALL PLTLP(PTITL,AT,ASAV,20,XTITL,YTITL,1,60.,XTITL1)
IF(II,EQ.2) CALL PLTLP(PTIT,AT,ASAV,20,XTITL,YTITL1,1,60.,XTITL1)
16 CONTINUE
C
C-----SET UP OFF-LINE PLOT
PLTYP=6
C
40 IF(ITAB,NE.1) GO TO 99
CALL PAGER(0)
CALL PAGER(8)
WRITE(LP,41)
WRITE(LP,42)
WRITE(LP,43)
DO 44 I=1,20
RMKG=AV(I)*DENL/1000.
EVRKG=AEV(I)/1000.
EVRLB=AEV(I)/454.
RMLB=AV(I)*DENL/454.
CALL PAGER(1)
WRITE(LP,45) AT(I),RMKG,RMLB,EVRKG,EVRLB
44 CONTINUE
GO TO 99
410 FORMAT (//9X,62HTABLE OF MASS REMAINING AND EVAPORATION RATE VS TI
IME - MODEL I//)
42 FORMAT(7X,4HTIME,9X,12HMASS REMAINS,5X,12HMASS REMAINS,7X,9HEVAP R
ATE,8X,9HEVAP RATE)
43 FORMAT(6X,6H(SECS),12X,4H(KG),13X,4H(LB),12X,8H(KG/SEC),9X,8H(LB/S
IEC)//)
45 FORMAT(3X,G12.4,4(5X,G12.4))
46 FORMAT(//1X,63HTHE FOLLOWING PARAMETERS ARE ESTIMATED IN CASE MODE
1L J FOLLOWS-)
100 FORMAT(/69H WARNING- THE CHEMICAL SPILLED HAS A LIQUID DENSITY SO
*CLOSE TO WATER/,66H THAT IT MAY OR MAY NOT SINK. FOR MODEL 1, IT W
*ILL BE ASSUMED THAT/,35H IT HAS A DENSITY OF 1.01 GM/CM**3.//)
99 CONTINUE
CALL TRACE(1,8,1)
END
SUBROUTINE EVDRP(VI,DENL,SURT, XLAT,TINF,TCRY,TIME,V,TOTEV,TIMCR)
C
C *** THIS ROUTINE CALCULATES THE RATE OF EVAPORATION AND TIME TO
C *** COMPLETELY EVAPORATE FOR A HEAVIER-THAN-WATER, INSOLUBLE
C *** LIQUID WITH A BOILING POINT LESS THAN THE AMBIENT TEMPERATURE.
C *** AT TIME=TIME
C
C *****
C *****INPUT ARGUMENTS *****
C *** VI = VOLUME OF THE SPILL OF THE LIQUID CM**3
C *** SURT % SURFACE TENSION OF LIQUID DYNE/CM
C *** XLAT = LATENT HEAT OF VAPORISATION OF LIQUID CAL/GM
C *** TINF = TEMPERATURE OF THE AMBIENT WATER DEG C
C *** TCRY = TEMPERATURE OF THE LIQUID SPILLED DEG C
C *** TIME = TIME(FROM THE INSTANT OF SPILL) AT WHICH SECS
C *** THE INFORMATION ABOUT EVAPORATION RATE IS TO BE KNOWN.
C *** DENL = DENSITY OF LIQUID SPILLED GM/CM3
C ***** OUTPUT ARGUMENTS *****
C *** V = VOLUME OF THE LIQUID LEFT IN THE SYSTEM CM**3
C *** TOTEV = TOTAL EVAPORATION RATE FROM THE LIQUID GMS/SEC
C *** AT TIME INSTANT =TIME.
C *** TIMCR - TIME TO EVAPORATE ALL OF THE LIQUID SECS
C
C ***** OTHER PARAMETERS *****
C *** WC = CRITICAL WEBER NUMBER AT WHICH LIQUID
C *** BREAKS UP (8)WC)10)
C *** CD = DRAG COEFFICIENT DURING THE DESCENT OF A

```

```

C *** DROPLET IN WATER.
C *** GEFF = EFFECTIVE GRAVITY ((GR*(DENL/DENW-1.)) CM/SEC**2
C *** GR = GRAVITATIONAL ACCELERATION CM/SEC**2
C *** PRW = PRANDTL NUMBER FOR WATER

DATA DENW,VISW,CPW,PRW,WC,CD,GR /1.0,0.01,1.0,7.0,10.0,0.4,980./
PI=3.141592654
A=1.778
DELRO=DENL-DENW
GEFF=GR*(DENL/DENW-1.)
DEL=0.056*WC
F=SQRT((1.-DEL)*((1.+A*(1.-DEL)**1.5)/(1.+A)))
RO=SQRT((3./8.)*(WC/F**2.)*CD*(SURT/(GR*DELRO)))
UO=F*SQRT((8./3.)*(GEFF/CD)*RO)
B=CPW*(TINF-TCRY)/XLAT
G=DENW*UO
REY=(UO**2.*RO)/(VISW/DENW)
STANT=0.69*(1./REY**0.3)*(1./PRW**0.66667)
CEVAP=G*STANT*ALOG(1.+B)
EVAPO=(3./RO)*VI*CEVAP
CT=DENL*RO/CEVAP
C *** CEVAP IS THE EVAPORATION RATE PER UNIT AREA AT ZERO TIME FROM A
C *** CT IS THE CHATACTERISTIC EVAPORATION TIME AS DEFINED
C *** EVAPO IS THE TOTAL EVAPORATION RATE IN GMS/SEC AT ZERO TIME.
TIMCR=CT/0.95
IF (TIME-TIMCR) 10,20,20
10 TOW=TIME/CT
ETA=(1.-0.95*TOW)**(1./0.95)
TOTEV=EVAPO*ETA**2.05
V=VI*ETA**3
RETURN
20 U=0.
TOTEV=0.0
RETURN
END
OVERLAY(10,2)
PROGRAM LINK1

C
C
C PROGRAM LINK1 IS A MAIN PROGRAM FOR OVERLAY 8, SEGMENT 2
C AND IS USED SIMPLY TO PASS CONTROL TO THE RATE MODELS K AND P.
C
COMMON/OVCNT/MODNO,OVLST(29),SGLST(29)
INTEGER OVLST,SGLST
C
CALL TRACE(0,8,2)
IF(MODNO.EQ.11) CALL MODK
IF(MODNO.EQ.13) CALL MODP
CALL TRACE(1,8,2)
END
SUBROUTINE BBLDIS(T,H,DIFCO,C,AM,FV,FM)
C
C *****
C
C THIS SUBROUTINE CALCULATES THE RATE OF DISSOLUTION OF A
C VAPOR BUBBLE IN WATER WHEN THE VAPOR IS RELEASED UNDER
C WATER. THE OUTPUT FROM THE ROUTINE IS THE FRACTION OF
C INITIAL VOLUME OF THE BUBBLE THAT ESCAPE INTO THE ATMOSPHERE
C FOR A GIVEN RELEASE DEPTH.
C
C INPUTS
C
C T = TEMPERATURE OF WATER. DEG.C
C H = DEPTH OF RELEASE UNDER WATER CMS
C DIFCO= DIFFUSION COEFFICIENT OF GAS SPECIMEN CM**2/SEC
C C = EQUILIBRIUM INTERFACIAL CONCENTRATION OF GAS IN WATER. G-MOLES/CM**3
C (EQUIVALENTLY IT IS THE SOLUBILITY OF GAS IN WATER) GM-MOLE/GM
C AM = MOLECULAR WEIGHT OF THE GAS
C
C OUTPUTS

```

FV = FRACTION OF THE INITIAL VOLUME OF THE GAS
EMERGING FROM THE SURFACE OF THE WATER BODY.
FM = FRACTION OF THE INITIAL MASS OF GAS
RELEASED UNDER WATER, EXCAPING INTO THE
ATMOSPHERE.

OTHER PARAMETERS

ALPHA= A DRAG FACTOR GENERALLY EQUAL TO ABOUT 30.
KFP = TURBULENT FLUCTUATION VALUE FOR DRAG
DUE TO FLOW OF GAS INSIDE THE BUBBLE.
(ABOUT 30)
DENW = DENSITY OF WATER=1.0 G/CM**3
GR = GRAVITATIONAL ACCELERATION=980. CM/SEC**2
PA = ATMOSPHERIC PRESSURE=1.0133E6 DYNES/CM**2
NUW = KINEMATIC VISCOSITY OF WATER=1.0E-2 CM**2/SEC
RUNIV= UNIVERSAL GAS CONSTANT=8.314E7 ERG/GMMOLDEGK
F = MASS TRANSFER COEFFICIENT FACTOR WHICH
TAKES INTO ACCOUNT THE FACT THAT THE
TRANSFER COEFFICIENT VALUE IS REDUCED DUE TO
THE EFFECT OF THE NEIGHBORING GAS BUBBLES.
RHOVI= THE INITIAL MOLAR DENSITY OF THE VAPOR
AT THE POSITION OF RELEASE UNDER WATER.
RI = THE INITIAL RADIUS OF BUBBLE WHICH IS EQUAL
TO THE CRITICAL RADIUS FOR STABILITY.
NI = THE NUMBER OF MOLES OF GAS IN THE INITIAL
BUBBLE.
RE = THE INITIAL REYNOLDS NUMBER
SC = THE SCHMIDT NUMBER
SH = SHERWOOD NUMBER.
SURT = INTERFACIAL TENSION BETWEEN WATER AND GAS. DYNES/CM
KLI = THE LIQUID SIDE MASS TRANSFER COEFFICIENT
INITIALLY CM/SEC

REAL NUW,NI,KLI,KFP
DATA DENW,GR,PA,NUW,RUNIV,EPSILN,ALPHA,KFP/ 1.0,980.,1.0133E6,1.E
1-2,8.314E7,0.5,0.08,0.5/
SURT=70.0
PI=3.141592654
RHOW=DENW
F=(1.5*(1.-EPSILN))*((1./3.)/EPSILN

CALCULATION OF DIMENSIONLESS PARAMETERS

PSTAR=(RHOW*GR*H/PA)
P=PA*(1.+PSTAR)
DENV=P*AM/(RUNIV*(T+273.))
U=((4./ALPHA)*(GR*SURT**2/(DENW**2*NUW)))*(DENV/DENW)**0.2
RI=((3./KFP)*(DENW/DENV))**((1./3.)*(SURT/(DENW*U)))
NI=((4./3.)*PI*RI**3)*(DENV/AM)
SC=NUW/DIFCO
AI=4.*PI*RI*RI
RE =U*RI/NUW
SH=0.6*(RE**0.5)*(SC**((1./3.))
KLI=(SH*DIFCO/RI)*F
USTAR=(U/H)*(NI/(KLI*AI*C))
B=((1.+PSTAR)/(PSTAR*USTAR))
PHI =B+(1.-B)*SQRT(1.+PSTAR)
IF(PHI.LT.0.0) PHI=0.0
FV=PHI**2
FM=FV/(1.+PSTAR)
RETURN
END

FUNCTION CNSPL(IFLAG,X,Y,Z,T,W,D,UF,UT,TP,DEL,XK,E,TNT,ZMDOT,TOW)
*** THIS FUNCTION IS CALLED BY THE DLIN INTEGRATION SUBROUTINE.
THIS FUNCTION RETURNS THE VALUES OF THE INTEGRAND IN THE INTEGRAL
OBTAIN THE CONCENTRATION AT ANY POSITION AND TIME WHEN THE SPILL
CONTINUOUS.
PI=3.14159265
GO TO (10,20,30),IFLAG

```

C *** CONTINUOUS SPILL IN STILL WATER
10 IF(T-TOW) 11,11,12
11 CNSPL=0.
12 RETURN
12 F1=(2.*ZMDOT)/(4.*PI*E)**1.5
F2=EXP(-(X**2+Y**2+Z**2)/(4.*E*(T-TOW)))-XK*(T-TOW)/(T-TOW)**1.5
CNSPL=F1*F2
20 RETURN
C *** TIDAL RIVER AND ESTURINE CONTINUOUS SPILL MODELS
30 IF(T-TOW) 31,31,32
31 CNSPL=0.
32 RETURN
32 TT=T-TOW
SIG=2.*PI/TP
F1=ZMDOT/(W*D*SQRT(4.*PI*E))
F2=((X-UF*TT+(UT/SIG)*(COS(SIG*(T-DEL))-COS(SIG*(TOW-DEL))))/SQRT(4.*E*TT))**2-XK*TT
F3=EXP(-F2)/SQRT(TT)
CNSPL=F1*F3
32 RETURN
END
SUBROUTINE DILUN(IFLAG,ICOND,ZMAS,ZMDOT,X,Y,Z,TIME,DIFCO,D,W,A,UF,
1UT,TP,DEL,XK,XN,C)
C
C *****
C *** THIS SUBROUTINE GIVES THE CONCENTRATION OF A WATER MISCIBLE LIQUID
C SPECIFIED SPATIAL POINT AND GIVEN TIME, FOR SPILL IN LAKE, RIVER O
C ESTUARY. ALL THE LIQUID SPILLED IS ASSUMED TO GO INTO SOLUTION
C WATER. THE SAME PROGRAM CAN ALSO BE USED FOR DISPERSION OF SOLIDS
C ARE NEUTRALLY BUOYANT OR WHOSE SETTLING TIMES ARE LARGE COMPARED T
C TIMES.
C
C THIS SUBROUTINE CANNOT BE USED WITH ACCURACY FOR CONCENTRATION PRE
C FOR THOSE FLUIDS WHICH REACT WITH WATER OR WHOSE BOILING POINT IS
C THAN THAT OF THE AMBIENT TEMPERATURE.
C *****
C ***** INPUT ARGUMENTS *****
C *** IFLAG = FLAG INDICATING WHERE THE SPILL OCCURS. (1 FOR SPILL IN
C STILL WATER, 2 FOR NON TIDAL RIVER, 3 FOR TIDAL REGIONS
C *** ICOND = A FLAG WHICH SPECIFIES WHETHER THE SPILL IS CONTINUOUS O
C OF SHORT DURATION (INSTANTANEOUS) SPILL. THE VALUE OF I
C 0 FOR SHORT DURATION SPILL AND 1 FOR CONTINUOUS SPILL.
C *** ZMAS = TOTAL MASS OF LIQUID SPILLED GMS
C *** ZMDOT = RATE OF MASS SPILL (TO BE GIVEN ONLY IF ICOND=1) GMS
C *** X,Y,Z = CO ORDINATE POSITIONS AT WHICH THE CONCENTRATION IS NEED
C THE ORIGIN IS ON THE WATER SURFACE. FOR RIVER SPILLS TH
C X-DIRECTION IS IN THE DIRECTION OF FLOW AND Z- DIRECTIO
C DEPTHWISE. CHS
C *** TIME = TIME (COUNTED FROM INSTANT OF SPILL) AT WHICH THE CONCEN
C TION AT POINT X,Y,Z IS TO BE KNOWN. SE
C *** DIFCO = DIFFUSION COEFFICIENT FOR THE LIQUID IN WATER CM*
C *** D = MEAN RIVER DEPTH CHS
C *** W = MEAN RIVER WIDTH CHS
C *** A = Y-COORDINATE OF THE POINT OF SPILL ON THE WATER SURFACE CHS
C *** UF = STREAM VELOCITY ( TO BE GIVEN IF IFLAG =2 OR 3) CHS
C *** UT = TIDAL VELOCITY AMPLITUDE (FOR IFLAG = 3) CHS
C *** TP = TIDAL PERIOD SEC
C *** DEL = PHASE LAG-- ESSENTIALLY THE TIME FOR THE NEXT HIGH WATER
C SLACK FROM THE INSTANT OF SPILL. SEC
C *** XK = DECAY COEFFICIENT ( TO BE GIVEN ONLY IF THE POLLUTANT DE
C AS PER THE FIRST ORDER DECAY EQUATION). 1./
C *** XN = MANNING FACTOR OF ROUGHNESS FOR RIVERS NON-DIM
C
C ***** OUTPUT ARGUMENTS *****
C *** C = CONCENTRATION OF THE POLLUTANT GMS
C *****
C
EXTERNAL CNSPL
DIMENSION AUX(1)

```

```

PI=3.14159265
IJ=0
ISAV=ICOND
B=W/2.
T=TIME
IF(ICOND) 20,20,5
5 TMT=ZMAS/ZMDOT
C *** TMT IS THE TIME TO EMPTY [ZMAS] AT THE RATE OF [ZMDOT].
IF(IFLAG.EQ.2) GO TO 20
IF(T-5.*TMT) 20,10,10
10 IJ=1
ICOND=0
20 CONTINUE
IF(ICOND)50,50,100
50 CONTINUE
C *** INSTANTANEOUS SPILL FORMULAE *****
GO TO (60,70,80),IFLAG
C *** SPILL IN CALM AND STILL WATER *****
60 C=((2.*ZMAS)/(4.*PI*T*DIFCO)**1.5)*EXP(-(X**2+Y**2+Z**2)/(4.*DIFCO
1*T))*EXP(-XK*T)
IF(IJ.EQ.1) ICOND=ISAV
RETURN
C ***** SPILL INTO A NON TIDAL RIVER *****
70 CALL DISP(W,D,IFLAG,T,UF,UT,XN,TP,E,EX ,EY,EZ)
TC=B**2/EY
C *** NEAR FIELD APPROXIMATION ***
74 F1=(2.*ZMAS)/((4.*PI*T)**1.5*SQR(EX*EY*EZ))
F2=EXP(-(XK*T+(X-UF*T)**2/(4.*EX*T)))
F3=EXP(-(Y-A)**2/(4.*EY*T))+EXP(-(Y+A+W)**2/(4.*EY*T))
1+EXP(-(Y-W+A)**2/(4.*EY*T))
F4=EXP(-(Z**2/(4.*EZ*T))+EXP(-(Z-2.*D)**2/(4.*EZ*T)))
F4=F4*EXP(-(Z+2.*D)**2/(4.*EZ*T))
C=F1*F2*F3*F4
RETURN
C ***** SPILL INTO TIDAL REGIONS OF A RIVER *****
80 CALL DISP(W,D,IFLAG,T,UF,UT,XN,TP,E,EX ,EY,EZ)
SIG=2.*PI/TP
C *** THE CROSS SECTIONAL MEAN CONCENTRATION IS CALCULATED ASSUMING
C THE RIVER OSCILLATION VELOCITY TO BE SINUSOIDAL.
F1= ZMAS /(W*D*SQR(4.*PI*E*T))
F2=EXP(-XK*T)
F31=X-(UF*T)
F32=(UT/SIG)*(COS(SIG*(T-DEL))-COS(SIG*DEL))
F33=(F31+F32)/SQR(4.*E*T)
F33=ABS(F33)
IF(F33.GT.9.3) C=0.0
IF(F33.GT.9.3) RETURN
F3= . /EXP(F33*F33)
C=F1*F2*F3
IF(IJ.EQ.1) ICOND=ISAV
RETURN
C
C ***** CONTINUOUS SPILLS *****
C *** IN THE FOLLOWING PROGRAM ON THE CONTINUOUS SPILLS WE ASSUME THAT
C ***RATE OF SPILL [ZMDOT] IS A CONSTANT.
10 GO TO (110,120,130),IFLAG
C ** CONTINUOUS SPILL IN A STILL WATER REGION *****
110 EPS=0.1
NDIM=25
IF(T-TMT) 112,112,113
112 CALL DLIN(IFLAG,X,Y,Z,T,UF,UT,TP,DEL,W,D,XK,DIFCO,TMT,ZMDOT,0.0,
1T,EPS,NDIM,CNSPL,C,IER,AUX)
RETURN
113 CALL DLIN(IFLAG,X,Y,Z,T,UF,UT,TP,DEL,W,D,XK,DIFCO,TMT,ZMDOT,0.0,
1TMT,EPS,NDIM,CNSPL,C,IER,AUX)
RETURN
C *** SPILL IN A NON TIDAL RIVER *** WE ASSUME THAT THE LONGITUDINAL
C SION IS SMALL. THE CONCENTRATION GIVEN IS THE CROSS SECTIONAL AVE
120 C=0.
IF(X-UF*T) 121,121,126
121 CALL DISP(W,D,IFLAG,T,UF,UT,XN,TP,E,EX ,EY,EZ)
TC=B**2/EY

```

```

XC=UF*TC
IF(X-0.3*XC) 122,122,123
C **** NEAR FIELD APPROXIMATION
122 F1=ZMDOT/(2.*PI*X*SQRT(EY*EZ))
TCN=X/UF
F2=EXP(-((Y-A)**2/(4.*EY*TCN)))+EXP(-((Y+A+W)**2/(4.*EY*TCN)))
1+EXP(-((Y-W+A)**2/(4.*EY*TCN)))
F3=EXP(-(Z**2/(4.*EZ*TCN)))+EXP(-((Z-2.*D)**2/(4.*EZ*TCN)))
F4=EXP(-(XK*TCN))
C=F1*F2*F3*F4
TSAV=(X/UF)+TMT
IF(T.GT.TSAV) C=0.0
RETURN
C *** FAR FIELD APPROXIMATION *****
123 OMG=SQRT(UF**2+4.*XK*E)
F1=ZMDOT/(W*D*OMG*2.)
G1=1.
G2=1.
IF(T-TMT) 125,125,124
124 G1=ERF((X+OMG*(T-TMT))/SQRT(4.*E*(T-TMT)))
G2=ERF((X-OMG*(T-TMT))/SQRT(4.*E*(T-TMT)))
125 F2=(ERF((X+OMG*T)/SQRT(4.*E*T))-G1)*EXP((X*0.5/E)*(UF+OMG))
F3=(ERF((X-OMG*T)/SQRT(4.*E*T))-G2)*EXP((X*0.5/E)*(UF-OMG))
C=F1*(F2-F3)
126 RETURN
C **** CONTINUOUS INJECTION IN TIDAL RIVERS *****
130 CALL DISP(W,D,IFLAG,T,UF,UT,XN,TP,E,EX,EY,EZ)
131 ESP=0.01
NDIM=25
IF(T-TMT) 132,132,133
132 CALL DLIN(IFLAG,X,Y,Z,T,UF,UT,TP,DEL,W,D,XK,E ,TMT,ZMDOT,0.0,
1T,EPS,NDIM,CNSPL,C,IER,AUX)
RETURN
133 CALL DLIN(IFLAG,X,Y,Z,T,UF,UT,TP,DEL,W,D,XK,E ,TMT,ZMDOT,0.0,
1TMT,EPS,NDIM,CNSPL,C,IER,AUX)
RETURN
END
SUBROUTINE DLIN(IFLAG,XX,YY,ZZ,T,UF,UT,TP,DEL,W,D,XK,S,TMT,ZMDOT,
1XL,XU,EPS,NMAX,FCT,Y,IER,AUX)
C
C THIS ROUTINE PERFORMS AN INTEGRATION OF A CONCENTRATION EQUATION
C WHICH GIVES THE CONCENTRATION AT SOME DOWNSTREAM LOCATION FOR
C CONTINUOUS SPILLS OF A CHEMICAL INTO A TIDAL REGION.
C
C THE CODING USED WAS TAKEN ALMOST VERBATIM FROM A SAMPLE PROGRAM
C GIVEN IN THE BOOK - APPLIED NUMERICAL METHODS - BY CARNAHAN,
C LUTHER, AND WILKES PUBLISHED BY JOHN WILEY AND SONS, INC. SEE
C THE SUBROUTINE NAMED TROMB ON PAGE 96.
C
C DIMENSION TA(20,20),AUX(1)
NMAX=10
IF(XU.GT.6000.) NMAX=11
C NMAX IS AN INTEGER WHICH DEFINES THE ACCURACY OF THIS INTEGRATION.
C IT WAS ARBITRARILY SET TO THE VALUES SHOWN TO PROVIDE A REASONABLE
C DEGREE OF ACCURACY WHILE USING A REASONABLE AMOUNT OF COMPUTATION
C TIME. THIS COMPROMISE CAN, HOWEVER, SOMETIMES RESULT IN CURVES
C WHICH CONTAIN ONE OR MORE ERRATIC POINTS.
C JMAX=NMAX+1
H=XU-XL
TA(1,1)=0.5*H*(FCT(IFLAG,XX,YY,ZZ,T,W,D,UF,UT,TP,DEL,XK,S,TMT,
1ZMDOT,XL)+FCT(IFLAG,XX,YY,ZZ,T,W,D,UF,UT,TP,DEL,XK,S,TMT,ZMDOT,XU)
2)
DO 2 N=1,NMAX
TA(N+1,1)=0.0
FR=H/2.0**N
IMAX=2**N-1
DO 1 I=1,IMAX,2
TS=FLOAT(I)*FR+XL
1 TA(N+1,1)=TA(N+1,1)+FCT(IFLAG,XX,YY,ZZ,T,W,D,UF,UT,TP,DEL,XK,S,
1TMT,ZMDOT,TS)
2 TA(N+1,1)=TA(N,1)/2.0+H*TA(N+1,1)/2.0**N
DO 3 J=2,JMAX

```

```

NXM=NMAX-J+2
FORJ=4.0*(J-1)
DO 3 N=1,NXM
3 TA(N,J)=(FORJ*TA(N+1,J-1)-TA(N,J-1))/(FORJ-1.0)
Y=TA(1,JMAX)
RETURN
END
SUBROUTINE MODK

```

THE DECISION PROCESSES WITHIN SUBROUTINE MODK ARE EXCEEDINGLY COMPLEX. A DETAILED DISCUSSION IS THEREFORE PRESENTED IN THE FOLLOWING TO CLARIFY THE VARIOUS BRANCHINGS WHICH TAKE PLACE DURING ITS EXECUTION.

MODEL K IS USED TO ASSESS THE HAZARDS OF SLIGHTLY SOLUBLE CHEMICALS WHICH HAVE A BOILING POINT LESS THAN THE AMBIENT TEMPERATURE. WHEN SUCH A SUBSTANCE IS OF FINITE SOLUBILITY AND IS RELEASED UNDER WATER, SUBROUTINE BBLDIS IS USED TO DETERMINE THE AMOUNT OF THE SUBSTANCE DISCHARGED WHICH GOES INTO SOLUTION AND THE AMOUNT WHICH IS EVOLVED AS VAPOR FROM THE SURFACE OF THE WATER. THE PORTION WHICH GOES INTO SOLUTION IS THEN USED IN MODEL P TO DETERMINE WATER POLLUTION HAZARDS. THE AMOUNT OF VAPOR IS TRANSFERRED TO MODELS M AND N, AND SUBSEQUENTLY TO MODELS C1 AND C2, FOR THE DETERMINATION OF VAPOR DISPERSION HAZARD EXTENTS. SUBROUTINE BBLDIS IS APPROPRIATE FOR USE ONLY IF THE CHEMICAL IS RELEASED AT A DEPTH GREATER THAN ABOUT TEN FEET UNDER WATER AND IS SLIGHTLY OR MODERATELY SOLUBLE IN WATER. WHEN THE SUBSTANCE IS RELEASED ON OR NEAR THE SURFACE OF THE WATER BODY AND/OR IS COMPLETELY MISCIBLE IN WATER, BBLDIS CANNOT BE USED. IN THESE CASES, IT IS ASSUMED FOR THE PURPOSE OF EXECUTION OF MODEL P, THE WATER DISPERSION MODEL, THAT ALL THE LIQUID GOES INTO SOLUTION. THE AMOUNT OF VAPOR EVOLVED IS ESTIMATED IN MODEL N, THE VAPOR DISPERSION MODEL.

THE DECISION PROCESS IS COMPLICATED BY TWO FACTORS. ONE OF THESE CONCERNS THE PROPERTY FILE TAPE SINCE IT LISTS THE SOLUBILITY DATA FOR A SUBSTANCE AS MISSING IF THE CHEMICAL IS COMPLETELY MISCIBLE IN WATER. THIS WAS DONE BECAUSE THE TRUE VALUE IS INFINITY AND CANNOT BE PROPERLY REPRESENTED. ITS CONSEQUENCE IS THAT ACCESS TO THIS DATA DOES NOT ALLOW ACCURATE DETERMINATION OF WHETHER THE SUBSTANCE IS COMPLETELY MISCIBLE OR IS OF UNKNOWN FINITE SOLUBILITY. TO ACCOUNT FOR THIS PREDICAMENT, THE MODEL ASSUMES THE SUBSTANCE IS COMPLETELY MISCIBLE IN WATER AND PROCEEDS AS DESCRIBED ABOVE.

THE OTHER PROBLEM ARISES FROM THE DENSITY OF THE SUBSTANCE. IF THE DENSITY IS GREATER THAN WATER, THE SUBSTANCE IS RELEASED ON OR NEAR THE BOTTOM OF THE WATERBODY, AND THE WATER DEPTH IS SUCH THAT HYDROSTATIC PRESSURE DOES NOT ALLOW THE CHEMICAL TO BOIL (CASE 1), VAPOR BUBBLES WILL NOT REACH THE SURFACE AND ALL THE CHEMICAL WILL EVENTUALLY DISSOLVE. THE DEPTH AT WHICH THE LIQUID WILL NOT BOIL IS CALLED ITS CRITICAL DEPTH. IF THE SUBSTANCE IS DENSER THAN WATER AND IS RELEASED AT A DEPTH LESS THAN ITS CRITICAL DEPTH IN A WATERBODY WHICH IS DEEPER THAN ITS CRITICAL DEPTH (CASE 2), SOME OF THE SUBSTANCE WILL BOIL OFF AND FORM VAPOR BUBBLES WHICH HEAD TOWARDS THE SURFACE WHILE THE REST SINKS TO A DEPTH AT WHICH IT CANNOT BOIL. IF THE CHEMICAL IS LESS DENSE THAN WATER AND IS RELEASED AT ANY POINT UNDER WATER (CASE 3) IT WILL EITHER IMMEDIATELY BEGIN TO BOIL OR RISE UNTIL IT REACHES A DEPTH AT WHICH IT CAN BOIL.

FOR CASE 1, MODEL K ASSUMES THAT ALL THE CHEMICAL INSTANTLY DISSOLVES IN THE WATER AND EXECUTES MODEL P. THE USERS MANUAL NOTES THAT THE USER MAY WISH TO EXECUTE MODEL X FOR MORE ACCURATE ANSWERS IF THE SUBSTANCE IS ONLY SLIGHTLY SOLUBLE.

FOR CASE 2, IT IS ASSUMED THAT THE CHEMICAL NEITHER RISES NOR SINKS. ALTHOUGH THIS ASSUMPTION TENDS TO OVERESTIMATE THE AMOUNT OF VAPOR WHICH REACHES THE SURFACE AND UNDERESTIMATE THE AMOUNT WHICH DISSOLVES, A BETTER PROCEDURE COULD NOT BE ENVISIONED.

FOR CASE 3, IT IS ASSUMED THAT THE CHEMICAL IS RELEASED AT A DEPTH JUST ABOVE ITS CRITICAL DEPTH. THIS ALLOWS THE CHEMICAL TO BOIL JUST AS IT WILL WHEN IT RISES TO THIS DISTANCE BENEATH THE SURFACE.

```

C
1 DATA MOD/4H K /
  CONTINUE
  LP=6
  IR=0
  IS=6
  C=0.1

OBTAIN NECESSARY DATA
CALL BEGPR(MOD)
CALL FRCL(2021,H,IS,IR)
B=10.*12.*2.54
IF(H.LT.B) GO TO 5
CALL FRCL(1002,AM,IS,IR)
CALL FRCL(1003,TB,IS,IR)
CALL FRCL(1010,AVP,IS,IR)
CALL FRCL(1011,BVP,IS,IR)
CALL FRCL(1012,CVP,IS,IR)
CALL FRCL(1021,DENL,IS,IR)
CALL FRCL(1025,TC,IS,IR)
CALL FRCL(2004,TEMP,IS,IR)
CALL FRCL(2007,SPANT,IS,IR)
CALL FRCL(2023,T,IS,IR)

CALCULATE DIFFUSION COEFFICIENT OF CHEMICAL IN WATER
CALL PAGER(4)
WRITE(LP,10)
CALL CDIFW(AM,DENL,T,TC,TB,DIFCO)
CALL FSV(2043,DIFCO,4)
CALL FRCL(2043,DIFCO,IS,IR)

CALL ROUTINE SOLUB TO FIND SOLUBILITY OF CHEMICAL AT THE WATER
TEMPERATURE
SOLUB CALLS DATA OF FIELD NUMBERS 1026,1028, AND 1029.
CALL SOLUB(T,CSAT,IS,IR)
C=CSAT/(100.*AM)
IF(C.NE.0.0) GO TO 5
CALL PAGER(5)
WRITE(LP,70)
IF(C.EQ.0.0) GO TO 5
5 CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 1
IF(H.LT.B) GO TO 30
IF(C.EQ.0.0) GO TO 31
TMX=TEMP
IF(T.GT.TMX) TMX=T
CALL CRITD(AVP,BVP,CVP,TMX,DEPTH)
CALL OUTPR(MOD)
IF(DENL.GE.1.0) GO TO 6
IF(H.LT.DEPTH) GO TO 6
CALL PAGER(5)
WRITE(LP,80)
6 IF(DENL.LT.1.0.AND.H.GE.DEPTH) H=DEPTH
IF(H.LT.B) GO TO 8
CALL BBLDIS(T,H,DIFCO,C,AM,FV,FM)
IF(DEPTH.LT.H) FM=0.0
IF(FM.NE.0.0) GO TO 7
CALL PAGER(4)
WRITE(LP,20)
7 AMTUP=FM*SPANT
CALL ISV(2029,0,4)
CALL ISV(2084,0,4)
CALL FSV(4001,AMTUP,4)
AMTLQ=SPANT-AMTUP
CALL FSV(4002,AMTLQ,4)
VOL=AMTLQ/DENL
CALL FSV(4003,VOL,4)
CALL FSV(4046,DEPTH,4)
CALL FSV(4068,TB,4)

```

```

CALL PAGER(4)
WRITE(LP,40)
GO TO 50
8 CALL PAGER(6)
WRITE(LP,90)
CALL ISV(2029,0,4)
CALL ISV(2084,1,4)
CALL FSV(4001,0,0,4)
CALL FSV(4002,SPAMT,4)
VOL=SPAMT/DENL
CALL FSV(4003,VOL,4)
CALL FSV(4046,DEPTH,4)
CALL FSV(4068,TB,4)
WRITE(LP,40)
GO TO 50
30 CALL PAGER(6)
WRITE(LP,60)
31 CALL PAGER(4)
CALL ISV(2084,1,4)
IF(C.EQ.0,0) CALL FSV(4001,0,0,4)
IF(C.EQ.0,0) CALL FSV(4002,SPAMT,4)
VOL=SPAMT/DENL
IF(C.EQ.0,0) CALL FSV(4003,VOL,4)
WRITE(LP,40)
50 CALL MODP
CALL ENDPR(MOD)
99 RETURN
10 FORMAT(/66H THE DIFFUSION COEFFICIENT OF THE CHEMICAL IN WATER IS
1CALCULATED./)
20 FORMAT(/64H THE CHEMICAL IS RELEASED AT TOO GREAT A DEPTH FOR VAPO
1R BUBBLES/22H TO REACH THE SURFACE./)
40 FORMAT(/53H MODEL P IS EXECUTED TO DETERMINE THE WATER POLLUTION/
154H HAZARDS OF THE CHEMICAL WHICH DISSOLVES IN THE WATER./)
60 FORMAT(/62H SINCE THE RELEASE IS ON OR NEAR THE SURFACE, AND A MOD
1EL DOES/59H NOT EXIST WHICH ESTIMATES THE PROPORTION OF CHEMICAL W
2HICH/61H DISSOLVES, IT IS ASSUMED THAT IT ALL DISSOLVES FOR EXECUT
3ION/12H OF MODEL P./)
70 FORMAT(/54H SUFFICIENT DATA DOES NOT EXIST TO ALLOW DETERMINATION/
151H OF WHETHER THE CHEMICAL IS FULLY MISCIBLE IN WATER/54H OR OF F
2INITE SOLUBILITY. IT IS THEREFORE ASSUMED THAT/35H ALL THE LIQUID
3RELEASED DISSOLVES./)
80 FORMAT(/58H SINCE THE CHEMICAL IS LESS DENSE THAN WATER AND WILL R
1ISE/54H TO A DEPTH AT WHICH IT CAN BOIL, THE DEPTH OF RELEASE/38H
3WAS CHANGED TO BE ITS CRITICAL DEPTH./)
90 FORMAT(/65H SINCE THE CRITICAL DEPTH IS LESS THAN 10 FEET DEEP, TH
1E SPECIFIC/69H AMOUNT OF CHEMICAL WHICH DISSOLVES IN THE WATER CAN
2NOT BE ESTIMATED./67H IT IS THEREFORE ASSUMED THAT ALL LIQUID DISS
3OLVES FOR EXECUTION OF/25H WATER POLLUTION MODEL P./)
END
SUBROUTINE MODP

```

C
C
C
C
C

SUBROUTINE MODP OBTAINS THE NECESSARY DATA FOR EXECUTION OF
SUBROUTINE DILUN, WHICH CALCULATES THE CONCENTRATION OF A
WATER-MISCIBLE LIQUID AT ANY SPECIFIED SPATIAL POINT AND
GIVEN TIME FOR A SPILL IN A LAKE, RIVER, OR ESTUARY.

```

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
LOGICAL YESNO
DIMENSION ASAV(40),ASAVT(40)
DIMENSION AT(40),AC(40),AXD(40),ATIM(40),ACONC(40,2)
DIMENSION PTITL(6),XTITL(6),XTITL1(6),YTITL(6)
EQUIVALENCE (XBX(61),X),(XBX(62),Y),(XBX(63),Z)
EQUIVALENCE (XRX(66),UX),(XBX(67),TMINX),(XBX(68),ACONC(1,1))
DATA MOD/4H P/
ODATA (PTITL(I),I=1,6)/8HCONCENTR,8HATION VS,8H TIME AT,
18H A FIXED,8H POINT -,8H MODEL P/
ODATA (XTITL(I),I=1,6)/8HELAPSED,8HTIME FRO,8HM START,
18HOF SPILL,8H.....(8HMINUTES)/
ODATA (XTITL1(I),I=1,6)/8HELAPSED,8HTIME FRO,8HM START,
18HOF SPILL,8H.....(8H.HOURS)/
ODATA (YTITL(I),I=1,6)/8H CONCE,8HNTRATION,8H AT P,

```

```

18H0INT XYZ,1H ,5H(PPH)/
4 CONTINUE
IR=0
IS=6
LP=6

```

C
C
C

OBTAIN NECESSARY DATA

```

CALL BEGPR(MOD)
CALL IRCL(2028,IFLAG,IS,IR)
CALL IRCL(2029,ICOND,IS,IR)
IF(IFLAG.EQ.1) CALL FRCL(1002,AM,IS,IR)
IF(IFLAG.EQ.1) CALL FRCL(1003,TBOIL,IS,IR)
CALL FRCL(1004,DENL,IS,IR)
IF(IFLAG.EQ.1) CALL FRCL(1021,DENLB,IS,IR)
IF(IFLAG.EQ.1) CALL FRCL(1025,TCRIT,IS,IR)
IF(IFLAG.EQ.1) CALL FRCL(2023,TWAT,IS,IR)
CALL FRCL(2039,X,IS,IR)
CALL FRCL(2040,Y,IS,IR)
CALL FRCL(2041,Z,IS,IR)
CALL FRCL(2042,TIME,IS,IR)
IF(IFLAG.NE.1) CALL FRCL(2044,D,IS,IR)
IF(IFLAG.NE.1) CALL FRCL(2045,W,IS,IR)
IF(IFLAG.NE.1) CALL FRCL(2046,A,IS,IR)
IF(IFLAG.NE.1) CALL FRCL(2047,UF,IS,IR)
IF(IFLAG.NE.3) GO TO 100
CALL FRCL(2048,UT,IS,IR)
CALL FRCL(2049,TP,IS,IR)
CALL FRCL(2050,DEL,IS,IR)
100 CONTINUE
CALL FRCL(2051,XK,IS,IR)
IF(IFLAG.NE.1) CALL FRCL(2052,XN,IS,IR)
CALL FRCL(4002,ZMAS,IS,IR)
IF(ICOND.EQ.1) CALL FRCL(4049,ZMDOT,IS,IR)

```

C
C
C

CALCULATE DIFFUSION COEFFICIENT OF CHEMICAL IN WATER.

```

IF(IFLAG.NE.1) GO TO 30
CALL PAGER(3)
WRITE(6,40)
CALL CDIFW(AM,DENLB,TWAT,TCRIT,TBOIL,DIFCO)
DIFCO=DIFCO*1000.0
CALL FSV(2043,DIFCO,4)
CALL FRCL(2043,DIFCO,IS,IR)
30 CONTINUE
CALL IRCL(3008,IPPF,IS,IR)
CALL IRCL(3015,ITAB,IS,IR)
IF(ITAB.EQ.0.AND.IPPF.EQ.0) GO TO 1
CALL FRCL(2037,TMXP,IS,IR)
1 CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 4
CALL DILUN
CALL DILUN(IFLAG,ICOND,ZMAS,ZMDOT,X,Y,Z,TIME,DIFCO,D,W,A,UF,
1 UT,TP,DEL,XK,XN,CNTR)

```

C
C

C
C
C

UPDATE DATA BASE

```

CALL OUTPR(MOD)
IF(CNTR.GT.DENL) CNTR=DENL
CALL FSV(4022,CNTR,4)
CS=CNTR*1000000.0
CPPM=CS/DENL
CALL PAGER(2)
WRITE(LP,31) CPPM,CS
CALL ENDPR(MOD)

```

C
C
C
C

CALCULATE DATA OF CONCENTRATION VS TIME AT FIXED POINT IF REQUESTED.

```

ISTOP=0

```

```

IF(IPPF.EQ.0.AND.ITAB.EQ.0) GO TO 99
IF(IPPF.EQ.2.AND.ITAB.EQ.2) GO TO 70
IF (IFLAG.EQ.2) GO TO 25
TMAXI=TMXP
TMINI=60.
GO TO 20
25 TMX=X/UF
RH=W*D/(2.*D+W)
USTAR=6.7305*XN*UF/RH*(1./6.)
DIST=7.*SQRT(4.*.0067*USTAR*RH*TMX)
TYM=DIST/UF
TMAXI=TMX+(TYM/2.)
TMINI=TMX-(TYM/2.)
IF(TMINI.LT.60.) TMINI=60.
20 DT=(TMAXI-TMINI)/39.
DO 10 I=1,40
AT(I)=(FLOAT(I-1)*DT)+TMINI
CALL DILUN(IFLAG,ICOND,7MAS,ZMDOT,X,Y,Z,AT(I),DIFCO,D,W,A,UF,
1 UT,TP,DEL,XK,XN,AC(I))
10 CONTINUE
IF(IFLAG.EQ.2) GO TO 13
IF(AC(37).GT.0.0) GO TO 13
I=36
IF(ISTOP.EQ.1) GO TO 13
12 IF(AC(I).GT.0.0) GO TO 11
I=I-1
IF(I.EQ.1) GO TO 13
GO TO 12
11 TMAXI=AT(I+1)
ISTOP=1
GO TO 20

C
C
C GENERATE PLOT OF CONCENTRATION VS TIME AT USER SPECIFIED POINT
13 CONTINUE
DO 15 I=1,40
ASAV(I)=AC(I)*1000000./DENL
IF(ASAV(I).GT.1000000.) ASAV(I)=1000000.
15 ASAV(I)=AT(I)/60.
IDC=0
IF(IPPF.EQ.3) IDC=1
IF(IPPF.EQ.1) IDC=1
IF(IDC.NE.1) GO TO 70
CALL PLTLP(PTITL,ASAV,ASAV,40,XTITL,YTITL,1,60.,XTITL1)

C
C-----SET UP OFF-LINE PLOT
PLTYP=7
II=0
JJ=20
UX=UF
TMJNX=TMINI
DO 16 I=1,40,2
II=II+1
JJ=JJ+1
XBX(II)=AT(I)
16 XBX(JJ)=AC(I)

C
C
C GENERATE TABLE OF CONCENTRATION VS TIME AT USER SPECIFIED POINT.
70 IF(ITAB.EQ.2) GO TO 80
IF(ITAB.EQ.0) GO TO 80
CALL PAGER(0)
CALL PAGER(2)
WRITE(6,71)
CALL PAGER(3)
WRITE(6,72)
CALL PAGER(2)
WRITE(6,73)
DO 74 I=1,40
CS=ASAV(I)*DENL
CALL PAGER(1)
WRITE(LP,75) ASAV(I),ASAV(I),CS

```

```

74 CONTINUE
  CALL PAGER(2)
C
C
C  CALCULATE DATA FOR A TABLE OF CONCENTRATIONS VS. TIME AND
  DISTANCE, AS APPROPRIATE, FOR A NON-TIDAL FLOWING RIVER.
80 IF(ITAB.LE.1.AND.IPPF.LE.1) GO TO 69
  IF(IFLAG.EQ.2) GO TO 5
  CALL PAGER(0)
  CALL PAGER(3)
  WRITE(LP,49)
  5 IF(IFLAG.NE.2) GO TO 99
  TMINI=60.
  YM=A
  ZM=0.0
  DT=(TMXP-TMINI)/39.
  XMINI=UF*TMINI
  DT=ABS(DT)
  DO 60 J=1,2
  DO 50 I=1,40
    ATIM(I)=TMINI+(FLOAT(I-1)*DT)
    AXD(I)=UF*ATIM(I)
    IF(ICOND.EQ.1) ATIM(I)=ATIM(I)+(ZMAS/(ZMDOT*2.))
C**ABOVE LINE INSERTED BY R.G. POTTS ON 3 NOV 1978
C**SETS TIME TO OBTAIN MAX CONCENTRATION AT LOC AXD(I)
    CALL DILUN(IFLAG,ICOND,ZMAS,ZMDOT,AXD(I),YM,ZM,ATIM(I),DIFCO,D,W,
      1A,UF,UT,TP,DEL,XK,XN,ACONC(I,J))
  50 CONTINUE
  ZM=D
  60 CONTINUE
C
C
C  GENERATE TABLE FROM DATA COMPUTED ABOVE
  CALL PAGER(0)
  CALL PAGER(4)
  IF(ICOND.EQ.0) WRITE(6,2000)
2000 FORMAT (/12X,53HTABLE OF CONCENTRATION VS TIME AND DISTANCE - MODE
1L P//)
  IF(ICOND.NE.0) WRITE(6,2010)
2010 FORMAT (/11X,44HTABLE OF CONCENTRATION VS DISTANCE - MODEL P//)
  CALL PAGER(4)
  IF(ICOND.EQ.0) WRITE(6,2020)
2020 FORMAT (1X,2(4X,10HDOWNSTREAM),6X,8HELAPSED ,2(2X,13HCONCENTRATION
1)/2(6X,8HDISTANCE),8X,4HTIME,7X,10HAT SURFACE,4X,11HON RIVERBED/
2 6X,8H(METERS),7X,6H(FEET),7X,9H(MINUTES),2(4X,10H(MG/LITER))//)
  IF(ICOND.NE.0) WRITE(6,2030)
2030 FORMAT (1X,2(4X,10HDOWNSTREAM),2X,2(2X,13HCONCENTRATION)/2(6X,8HDI
1STANCE),7X,10HAT SURFACE,4X,11HON RIVERBED/6X,8H(METERS),7X,6H(FEE
2T),4X,2(4X,10H(MG/LITER))//)
  DO 65 I=1,40
    XMET=AXD(I)/100.
    TMNS=ATIM(I)/60.
    XFT=AXD(I)/(2.54*12.)
    IF(ACONC(I,1).GT.DENL) ACONC(I,1)=DENL
    IF(ACONC(I,2).GT.DENL) ACONC(I,2)=DENL
    TOPC=ACONC(I,1)*1000000.
    ACONC(I,1)=TOPC/DENL
    BOTC=ACONC(I,2)*1000000.
    ACONC(I,2)=BOTC/DENL
    CALL PAGER(1)
    IF(ICOND.EQ.0) WRITE(6,2040) XMET,XFT,TMNS,TOPC,BOTC
2040 FORMAT (1X,5(4X,G10.4))
    IF(ICOND.NE.0) WRITE(6,2050) XMET,XFT,TOPC,BOTC
2050 FORMAT (1X,2(4X,G10.4),2X,2(4X,G10.4))
  65 CONTINUE
  CALL PAGER(5)
  WRITE(6,2060)
2060 FORMAT (/5X,63HNOTE: TABLE GIVES CONCENTRATIONS AT SURFACE AND ON
1RIVERBED FOR/5X,66HDISTANCES DOWNSTREAM FROM THE SPILL LOCATION.
2DOWNSTREAM DISTANCE/5X,65HIS GIVEN ALONG LINES ON THE SURFACE AND
3RIVERBED SHIFTED FROM THE/5X,37HRIVER CENTERLINE BY THE SPILL OFFS
4ET.)

```



```

WRITE(LP,200)
CALL COMPD(AM,TA,DLB,DIFVA)
CALL FSV(2053,DIFVA,4)
CALL FRCL(2053,DIFVA,IS,IR)
CALL FRCL(4002,ZHL,IS,IR)
C
C
C   CALCULATE DIFFUSION COEFFICIENT OF LIQUID CHEMICAL IN WATER
IF(IFLAG.EQ.2) GO TO 60
CALL PAGER(3)
WRITE(LP,70)
CALL CDIFW(AM,DLB,TW,TCRIT,TROIL,DIFLW)
DIFLW=1000.*DIFLW
CALL FSV(2043,DIFLW,4)
CALL FRCL(2043,DIFLW,IS,IR)
60 CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 2
C
C
C   CALL EVAMX - THEN UPDATE DATA BASE
CALL OUTPR(MOD)
CSTM=CSTMT
SMAX=0.0
IFLG=0
DO 30 I=1,2
IF(CSTM.EQ.0.0) IFLG=1
IF(CSTM.EQ.0.0) GO TO 35
CALL EVAMX(ZHL,XHOL,ACR,BCR,CCR,DIFVA,DENL,DIFLW,IFLAG,D,W,US,XN
*,TW,CSTM,ZMV,S,SIZMX,THAT)
IF(IFLG.EQ.0.AND.I.EQ.2) GO TO 31
CALL PAGER(2)
IF(IFLG.EQ.0) WRITE(LP,50)
IF(IFLG.EQ.1) WRITE(LP,40)
CALL FSV(4023,ZMV,4)
CALL FSV(4024,S,4)
CALL FSV(2019,SIZMX,4)
CALL FRCL(2019,SIZMX,IS,IR)
SMAX=SIZMX
FLOW=ZMV/THAT
CALL FSV(4044,FLOW,4)
CALL FSV(4045,THAT,4)
TTM=THAT
GO TO 30
31 CALL PAGER(2)
WRITE(LP,40)
CALL FSV(4056,ZMV,4)
CALL FSV(4057,S,4)
CALL FSV(4058,SIZMX,4)
FLOW=ZMV/THAT
CALL FSV(4059,FLOW,4)
CALL FSV(4060,THAT,4)
GO TO 30
35 CSTM=CSTM*F
30 CONTINUE
IF(SMAX.EQ.0.0) SMAX=SIZMX
CALL PAGER(2)
WRITE(LP,39)
SMAX=0.638*SMAX
CALL FSV(2019,SMAX,6)
CALL PAGER(1)
WRITE(LP,38)
IF(TTM.LT.600.) CALL ISV(2061,0,4)
IF(TTM.GE.600.) CALL ISV(2061,1,4)
CALL ISV(2018,2,4)
CALL FSV(4068,TW,4)
IF(TTM.GE.600.) GO TO 98
CALL FRCL(4023,ZMV,IS,IR)
CALL FSV(4001,ZMV,6)
98 CALL ENDPR(MOD)
GO TO 99
38 FORMAT(52H THE VAPOR SOURCE PARAMETERS ARE ESTIMATED AS BEING-)

```

```

39 FORMAT(/1X,49H SIZE IS SET TO MEAN SIZE IN CASE MODEL S FOLLOWS.)
40 FORMAT(1X,44H FOR THE LOWER FLAMMABLE LIMIT CONCENTRATION./)
50 FORMAT(1X,28H FOR THE TOXIC CONCENTRATION./)
70 FORMAT(/64H THE DIFFUSION COEFFICIENT OF CHEMICAL WITH WATER IS CALCULATED./)
100 FORMAT(66H WARNING- MODEL R DOES NOT TAKE INTO ACCOUNT TIDAL EFFECTS. IT IS /42H ASSUMED THAT SPILL IS IN NON-TIDAL RIVER./)
200 FORMAT(/59H THE DIFFUSION COEFFICIENT OF VAPOR WITH AIR IS CALCULATED./)
99 CALL TRACE(1,8,3)
END
SUBROUTINE EVAMX(ZML,XMOL,ACR,BCR,CCR,DIFVA,DENL,DIFLW,IFLAG,D,W,
*US,XN,TW,CSTM,ZMV,S,SIZMX,THAT)
*****
*** THIS SUBROUTINE CALCULATES THE TOTAL MASS OF VAPOR PRODUCED WHEN
WATER MISCIBLE- HIGH VAPOR PRESSURE LIQUID IS SPILLED ON WATER A
MIXES WITH THE WATER DUE TO TURBULENT DIFFUSION.

***** INPUT ARGUMENTS *****
*** ZML = MASS OF LIQUID SPILLED GMS
*** XMOL = MOLECULAR WEIGHT OF THE LIQUID SPILLED
*** ACR = CONSTANTS IN THE VAPOR PRESSURE EQUATION P=10**(ACR-BC
*** BCR =
*** CCR =
*** DIFVA = DIFFUSION COEFFICIENT OF VAPOR IN AIR AT AMBIENT TEMP
*** DENL = DENSITY OF LIQUID AT THE SPILL TEMPERATURE GM/C
*** DIFLW = DIFFUSION COEFFICIENT OF LIQUID IN WATER CM*
*** IFLAG = A FLAG INDICATING THE LOCATION OF SPILL (1=STILL WATER,
2=FLOWING, NON TIDAL RIVER
*** D = RIVER DEPTH (TO BE GIVEN ONLY IF IFLAG=2) CMS
*** W = RIVER WIDTH CMS
*** US = AVERAGE VELOCITY OF THE STREAM CM/
*** XN = STREAM ROUGHNESS FACTOR
*** TW = WATER TEMPERATURE DEG
*** CSTM = LIMITING VALUE OF THE MOLE FRACTION CONCENTRATION. THAT
CONTRIBUTION TO EVAPORATION FROM WATER SURFACE REGIONS
THIS CONCENTRATION IS NEGLIGIBLE.

***** OUTPUT ARGUMENTS *****
*** ZMV = MASS OF VAPOR LIBERATED GMS
*** S = MAXIMUM DISTANCE IN THE STREAM DIRECTION BEYOND WHICH T
CONCENTRATION IS EVERYWHERE LESS THAN 'CSTM'. CMS
*** SIZMX = MAXIMUM SIZE (RADIUS) OF THE SPREAD CMS
*** THAT = TIME AT WHICH CONCENTRATION EVERYWHERE LESS THAN CSTM SECS

*****
COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
DIMENSION FX(101),ZM(101)
EQUIVALENCE (XBX(1),FX(1))
PI=3.141592654
TP=0.
T=0.
UT=0.
VOLI=ZML/DENL
CALL HMTX(DIFVA,XMOL,VOLI,HMP)
GO TO (10,20),IFLAG
10 EX=DIFLW
EY=DIFLW
EZ=DIFLW
GO TO 30
20 CALL DISP(W,D,IFLAG,T,US,UT,XN,TP,E,EX,EY,EZ)
30 CALL VAPPR(ACR,BCR,CCR,TW,PVAP)
CSTAR=(CSTM/(1.-CSTM))*(XMOL/18.)

C
C
*** CALCULATION OF THE CHARACTERISTIC CONSTANTS *****
THAT=(1./(4.*PI))*(2.*ZML/(CSTAR*SQRT(EX*EY*EZ)))*(2./3.)
A=SQRT(4.*EX*THAT)

```

```

      B=SQRT(4.*EY*THAT)
      DZMCH=PI*A*B*HMP*PVAP
      ZMCHR =DZMCH *THAT
C
C ***  INTEGRATION PREPARATION  *****
C
      N=101
      FX(1)=0.
      DO 40 I=2,N
      TOW=FLOAT(I-1)/FLOAT(N-1)
      CMAX=CSTAR/TOW**1.5
C
C ***  'TOW' IS THE NON DIMENSIONAL TIME = TOME/THAT. CMAX IS THE MAXIM
C      CONCENTRATION AT ANY TIME IN GM/CM**3
C
      CMAXM=CMAX/(CMAX+XMOL/18.)
      IF(TOW-0.98) 50,50,60
50      CMENM=(CMAXM-CSTM)/ALOG(CMAXM/CSTM)
      GO TO 40
60      CMENM=CSTM
40      FX(I)=-1.5*TOW*CMENM*ALOG(TOW)
      FX(N)=0.
      DTOW=1./FLOAT(N-1)
C
C ***  SIMPSON'S RULE INTEGRATION  *****
C
      CALL QSF(DTOW,FX,ZM,N)
      ZMV=ZMCHR*ZM(N)
      S=US*THAT
      SIZMX=W/2.
      IF(SIZMX.GT.SQRT(EY*THAT)) SIZMX=SQRT(EY*THAT)
      RETURN
      END
      SUBROUTINE QSF(H,Y,Z,NDIM)
C*****
C ***  SIMPSON'S RULE INTEGRATION ROUTINE , FOR DETAILS SEE THE IBM MA
C
      DIMENSION Y(1),Z(1)
      HT=.3333333*H
      L1=1
      L2=2
      L3=3
      L4=4
      L5=5
      L6=6
      IF(NDIM-5)7,8,1
C
C 1 NDIM IS GREATER THAN 5. PREPARATIONS OF INTEGRATION LOOP
      SUM1=Y(L2)+Y(L2)
      SUM1=SUM1+SUM1
      SUM1=HT*(Y(L1)+SUM1+Y(L3))
      AUX1=Y(L4)+Y(L4)
      AUX1=AUX1+AUX1
      AUX1=SUM1+HT*(Y(L3)+AUX1+Y(L5))
      AUX2=HT*(Y(L1)+3.875*(Y(L2)+Y(L5))+2.625*(Y(L3)+Y(L4))+Y(L6))
      SUM2=Y(L5)+Y(L5)
      SUM2=SUM2+SUM2
      SUM2=AUX2-HT*(Y(L4)+SUM2+Y(L6))
      Z(L1)=0.
      AUX=Y(L3)+Y(L3)
      AUX=AUX+AUX
      Z(L2)=SUM2-HT*(Y(L2)+AUX+Y(L4))
      Z(L3)=SUM1
      Z(L4)=SUM2
      IF(NDIM-6)5,5,2
C
C 2 INTEGRATION LOOP
      DO 4 I=7,NDIM,2
      SUM1=AUX1
      SUM2=AUX2
      AUX1=Y(I-1)+Y(I-1)
      AUX1=AUX1+AUX1
      AUX1=SUM1+HT*(Y(I-2)+AUX1+Y(I))
      Z(I-2)=SUM1

```

```

      IF(I-NDIM)3,6,6
3  AUX2=Y(I)+Y(I)
   AUX2=AUX2+AUX2
   AUX2=SUM2+HT*(Y(I-1)+AUX2+Y(I+1))
4  Z(I-1)=SUM2
5  Z(NDIM-1)=AUX1
   Z(NDIM)=AUX2
   RETURN
6  Z(NDIM-1)=SUM2
   Z(NDIM)=AUX1
   RETURN
C  END OF INTEGRATION LOOP
C  7 IF(NDIM-3)12,11,8
   NDIM IS EQUAL TO 4 OR 5
8  SUM2=1.125*HT*(Y(L1)+Y(L2)+Y(L2)+Y(L2)+Y(L3)+Y(L3)+Y(L3)+Y(L4))
   SUM1=Y(L2)+Y(L2)
   SUM1=SUM1+SUM1
   SUM1=HT*(Y(L1)+SUM1+Y(L3))
   Z(L1)=0.
   AUX1=Y(L3)+Y(L3)
   AUX1=AUX1+AUX1
   Z(L2)=SUM2-HT*(Y(L2)+AUX1+Y(L4))
   IF(NDIM-5)10,9,9
9  AUX1=Y(L4)+Y(L4)
   AUX1=AUX1+AUX1
   Z(L5)=SUM1+HT*(Y(L3)+AUX1+Y(L5))
10 Z(L3)=SUM1
   Z(L4)=SUM2
   RETURN
C  NDIM IS EQUAL TO 3
11 SUM1=HT*(1.25*Y(L1)+Y(L2)+Y(L2)-.25*Y(L3))
   SUM2=Y(L2)+Y(L2)
   SUM2=SUM2+SUM2
   Z(L3)=HT*(Y(L1)+SUM2+Y(L3))
   Z(L1)=0.
   Z(L2)=SUM1
12 RETURN
END
SUBROUTINE VAPPR(A,B,C,T,PVAP)
C*****
C  THIS SUBROUTINE CALCULATES THE VAPOR PRESSURE OF ANY COMPOUND AT T
C  GIVEN TEMPERATURE. THE EQUATION USED IS SIMILAR TO THE CLAUSIUS CL
C  EQUATION, NAMELY  $P = 10^{(A-B/(T+C))}$ .
C*****
C  INPUT ARGUMENTS *****
C  *** A = CONSTANT IN THE VAPOR EQUATION
C  *** B = CONSTANT IN THE VAPOR EQUATION
C  *** C = CONSTANT IN THE VAPOR EQUATION
C  *** T = TEMPERATURE AT WHICH THE VAPOR PRESSURE IS TO BE KNOWN
C*****
C  OUTPUT ARGUMENTS *****
C  *** PVAP = VAPOR PRESSURE AT THE TEMPERATURE =T= . ATM
C*****
C  PVAP=(10.**((A-(B/(T+C)))))/760.
C  RETURN
C  END
C  OVERLAY(10,4)
C  PROGRAM MODT
C*****
C  SUBROUTINE MODT UTILIZES ROUTINES RLJSP, TSPRD, AND FTCON TO
C  CALCULATE THE CONDITIONS AFTER THE SPILL OF AN INSOLUBLE OR
C  SLIGHTLY SOLUBLE CHEMICAL WHICH IS LIGHTER-THAN-WATER AND HAS A
C  BOILING POINT GREATER THAN THE AMBIENT TEMPERATURE. SEE THOSE
C  ROUTINES FOR SPECIFIC CAPABILITIES.
C*****
C  COMMON/C/PLTYP, XB(150)
C  INTEGER PLTYP
C  DIMENSION PTITL(6), XTITL(6), XTITL1(6), YTITL(6),
1  PTITL1(6), YTITL1(6)

```

```

DIMENSION AT(20),AS(20),ASAVT(20),ASAVS(20),ASAV(20,2)
EQUIVALENCE (XBX(64),IDIMX),(XBX(81),AT(1)),(XBX(101),AS(1))
EQUIVALENCE (XBX(21),ASAVT(1)),(XBX(121),ASAVS(1))
DATA MOD/4H T /
ODATA (PTITL (I),I=1,6)/8HPOOL RAD,8HIUS/LENG,8MTH VS TI,
18HME - MOD,8HEL T ,1H /
ODATA (PTITL1(I),I=1,6)/8HCONCENTR,8HATION VS,8H TIME AT,
18H A FIXED,8H POINT -,8H MODEL T/
ODATA (XTITL (I),I=1,6)/8HELAPSED ,8HTIME FRO,8HM START ,
18HOF SPILL,8H.....(,8HMINUTES)/
ODATA (XTITL1(I),I=1,6)/8HELAPSED ,8HTIME FRO,8HM START ,
18HOF SPILL,8H.....,8H.(HOURS)/
ODATA (YTITL (I),I=1,6)/8HPOOL RAD,8HIUS ,8HOR LENGTH,
18HH ,8H(METERS),1H /
ODATA (YTITL1(I),I=1,6)/8H CONCE,8HNTRATION,8H AT P,
18HOINT XYZ,1H ,1H /
12 CONTINUE
CALL TRACE(0,8,4)
LP=6
IR=0
IS=6
INDC=2
PI=3.14159265

C
C
C      OBTAIN DATA
C
CALL BEGPR(MOD)
CALL IRCL(2086,MDTYP,IS,IR)
IF(MDTYP.EQ.1) GO TO 5

C
C
C      WHEN MDTYP IS 0, THE MODEL FOR POOL SIZE VS TIME IS EXECUTED.
C      WHEN MDTYP IS 1, THE MODEL FOR CONCENTRATION AT XYZT IS EXECUTED.
C      WHEN MDTYP IS 2, BOTH THE ABOVE MODELS ARE EXECUTED.
C
CALL IRCL(2058,INDC,IS,IR)
IF(INDC.GE.1) IDIM=2
CALL FRCL(1004,DENL,IS,IR)

C
C      THE DENSITY IS CHECKED TO ENSURE IT IS LESS THAN THAT OF WATER
C      IF(DENL.LT.1.) GO TO 2
CALL PAGER(5)
WRITE(LP,3)
DENL=0.99
2 CONTINUE

C
IF(INDC.EQ.0) CALL FRCL(1006,VISL,IS,IR)
IF(INDC.EQ.0) CALL FRCL(1031,SURT,IS,IR)
IF(INDC.GE.1) CALL FRCL(2008,DIA,IS,IR)
IF(INDC.EQ.0) CALL FRCL(2020,CHNLW,IS,IR)
IF(INDC.GE.1) CALL FRCL(4049,FLOW,IS,IR)
IF(INDC.GE.1) CALL FRCL(4050,ENDTH,IS,IR)
CALL FRCL(2056,TIME,IS,IR)
IF(INDC.GE.1) CALL FRCL(2059,HGT,IS,IR)
CALL FRCL(4002,SPAMT,IS,IR)
VOL=SPAMT/DENL
IF(MDTYP.EQ.0) GO TO 6
5 CALL FRCL(1002,AM,IS,IR)
CALL FRCL(1003,TBOIL,IS,IR)
IF(MDTYP.EQ.1) CALL FRCL(1004,DENL,IS,IR)

C
C      THE DENSITY IS CHECKED TO ENSURE IT IS LESS THAN THAT OF WATER
C      IF(MDTYP.NE.1.OR.DENL.LT.1.) GO TO 4
CALL PAGER(5)
WRITE(LP,3)
DENL=0.99
4 CONTINUE

C
IF(MDTYP.EQ.1.OR.INDC.EQ.1) CALL FRCL(1006,VISL,IS,IR)
CALL FRCL(1008,SURFC,IS,IR)
CALL FRCL(1021,DENLB,IS,IR)
CALL FRCL(1025,TCRIT,IS,IR)
IF(MDTYP.EQ.1.OR.INDC.EQ.1) CALL FRCL(2020,CHNLW,IS,IR)

```



```

IDIM=1
CALL RLJSP(IDIM,VOL,DENL,VISL,SURT,TIME,CHNLW,SIZE)
AREA=CHNLW*SIZE
THICK=VOL/AREA
IF(THICK,LT,0.01) SIZE=VOL/(0.01*CHNLW)
15 CALL OUTPR(MOD)
CALL PAGER(1)
IF(IDIM,EQ,2) WRITE(LP,31)
IF(IDIM,EQ,1) WRITE(LP,25)
CALL FSV(4025,SIZE,4)
DIAM=2.*SIZE
IF(IDIM,EQ,1) DIAM=SQRT(SIZE*CHNLW*4./3.14159)
CALL FSV(4007,DIAM,4)
CALL PAGER(4)
WRITE(LP,16)
50 IF(MDTYP,EQ,0) GO TO 30
C
C
C
CALL FTCON TO FIND CONCENTRATION AT USER SPECIFIED POINT
CALL FTCON(DW,WW,US,DIFCO,CSAT,SPAMT,DENL,VISL,SURFC,XN,T,X,Y,Z,
1 DISRT,DISTM,PLCNX,CONC)
IF(CONC,GT,DENL) CONC=DENL
IF(MDTYP,EQ,1) CALL OUTPR(MOD)
CALL FSV(4061,DISRT,4)
CALL FSV(4062,DISTM,4)
CALL FSV(4063,PLCNX,4)
CALL FSV(4064,CONC,4)
CONC=CONC*1000000.0
CPPM=CONC/DENL
CALL PAGER(2)
WRITE(LP,51) CPPM,CONC
30 CALL ENDPR(MOD)
C
C
C
INTERROGATE USER PLOT AND TABLE REQUEST FLAGS
IF(ITPF,EQ,0.AND.ITAB,EQ,0) GO TO 99
IF(ITPF,EQ,1.OR.ITPF,EQ,3) GO TO 8
IF(ITAB,EQ,1.OR.ITAB,EQ,5) GO TO 8
GO TO 1
9 DT=TMXT/20.0
IF(MDTYP,EQ,1) GO TO 1
C
C
C
SET UP LOOP FOR CALCULATION OF PLOT ARRAYS OF SIZE OF LIQUID
POOL VERSUS ELAPSED TIME
ISVI=0
IDM=2
DO 10 I=1,20
AT(I)=(FLOAT(I-1)*DT)+DT
IF(INDC,LE,0) GO TO 11
CALL TSPRD(DENL,DIA,HGT,FLOW,AT(I),AS(I))
IF(AT(I),GT,ENDTM.AND.AS(I),GT,SIZMX) AS(I)=SIZMX
11 CONTINUE
IF(INDC,LE,0) CALL RLJSP(IDM,VOL,DENL,VISL,SURT,AT(I),CHNLW,AS(I))
IF(INDC,EQ,1) GO TO 9
IF(IDM,EQ,2) AREA=AS(I)*AS(I)*PI
IF(IDM,EQ,1) AREA=CHNLW*AS(I)
THICK=VOL/AREA
IF(THICK,LT,0.01.AND.IDM,EQ,2) AS(I)=SQRT(VOL/(0.01*PI))
IF(THICK,LT,0.01.AND.IDM,EQ,1) AS(I)=VOL/(0.01*CHNLW)
9 CONTINUE
IF(ISVI,GT,0.OR.INDC,EQ,1) GO TO 10
DIAMT=2.*AS(I)
IF(DIAMT,GT,CHNLW) IDM=1
IF(DIAMT,GT,CHNLW) ISVI=I
10 CONTINUE
IF(ITAB,NE,1.AND.ITAB,NE,5) GO TO 26
C
C
C
WRITE TABLE OF POOL SIZE VS TIME
CALL PAGER(0)
CALL PAGER(4)

```



```

XMET=AS(I)/100.
XFT=XMET*3.281
TMNS=AT(I)/60.
THRS=TMNS/60.
CALL PAGER(1)
WRITE(LP,54) XMET,XFT,TMNS,THRS,ASAV(I,1),ASAV(I,2)
80 CONTINUE
CALL PAGER(5)
WRITE(LP,55)
IF(MDTYP.EQ.0) GO TO 99
90 IF(ITAB.LT.3.AND.ITPF.LT.2) GO TO 99

C
C
C
COMPUTE DATA FOR A TABLE OR PLOT OF CONCENTRATION VS TIME AT A
USER SPECIFIED POINT XYZ

TS=X/US
IF(TS.LE.0.0) TS=1.
TS1=TS
91 CALL FTCON(DW,WW,US,DIFCO,CSAT,SPANT,DENL,VISL,SURFC,XN,TS,X,Y,Z,
1 DISRT,DISTH,PLCNX,CONC)
CONC=CONC*1000000.0/DENL
IF(CONC.GT.10.) TS=TS-(TS1/50.)
IF(CONC.GT.10.0.AND.TS.LT.0.0) TS=0.01
IF(CONC.GT.10.0.AND.TS.EQ.0.01) GO TO 96
IF(CONC.GT.10.0) GO TO 91
IF(TS.EQ.TS1) TS=TS1-300.0
IF(TS.LT.0.0) TS=0.01
96 DT=2.*(TS1-TS)/19.
IF(DT.LT.3.16) DT=60./19.
DO 92 I=1,20
AT(I)=TS+(FLOAT(I-1)*DT)
CALL FTCON(DW,WW,US,DIFCO,CSAT,SPANT,DENL,VISL,SURFC,XN,AT(I),X,Y,
1 Z,DISRT,DISTH,PLCNX,AS(I))
92 CONTINUE

C
C
C
WRITE A TABLE OF CONCENTRATION VS TIME AT A USER SPECIFIED POINT.

IF(ITAB.LT.3) GO TO 94
CALL PAGER(0)
CALL PAGER(5)
WRITE(LP,56)
CALL PAGER(1)
WRITE(LP,57)
CALL PAGER(1)
WRITE(LP,58)
94 DO 93 I=1,20
AT(I)=AT(I)/60.
THRS=AT(I)/60.
IF(AS(I).GT.DENL) AS(I)=DENL
AS(I)=AS(I)*1000000.0/DENL
CMGL=AS(I)*DENL
IF(ITAB.LT.3) GO TO 93
CALL PAGER(1)
WRITE(LP,59) AT(I),THRS,AS(I),CMGL
93 CONTINUE
IF(ITAB.LT.3) GO TO 95
CALL PAGER(2)
WRITE(LP,61)
CALL FRCL(2039,X,IS,IR)
CALL FRCL(2040,Y,IS,IR)
CALL FRCL(2041,Z,IS,IR)
95 IF(ITPF.LT.2) GO TO 99

C
C
C
WRITE PLOT OF CONCENTRATION VS TIME AT USER SPECIFIED POINT.
CALL PLTLP(PTITL1,AT,AS,20,XTITL,YTITL1,1,60.,XTITL1)

C
C
C
-----SET UP FOR SECOND OFF-LINE PLOT
ITMP=PLTYP
IF(ITMP.EQ.0) PLTYP=9
IF(ITMP.EQ.8) PLTYP=10
DO 701 I=1,20

```

```

701 AT(I)=60.*AT(I)
    GO TO 99
30FORMAT(/5X,58HWARNING - THE DENSITY OF THIS SUBSTANCE HAS BEEN EST
1IMATED/5X,57HAS BEING THE SAME AS OR GREATER THAN THAT OF WATER. U
2NDER/5X,70HTHESE CONDITIONS, THE MODEL ASSUMES THAT ITS DENSITY IS
3 0.99 GM/CM**3./)
7 FORMAT(/65H THE DIFFUSION COEFFICIENT OF THE CHEMICAL IN WATER IS
1CALCULATED/)
16 FORMAT(/6X,54H*** THIS ANALYSIS ASSUMES THE POOL WILL STOP SPREADI
1NG/10X,39HWHEN ITS THICKNESS IS 0.01 CENTIMETERS./)
18 FORMAT(/6X,49H*** SINCE THE FLOW FROM THE TANK STOPS BEFORE THE/10
1X,49HTIME AT WHICH THESE POOL SIZES WERE COMPUTED, THE/10X,47HMODE
2L ASSUMES THE POOL STOPS SPREADING WHEN ITS/10X,30HTHICKNESS IS 0.
301 CENTIMETERS./)
21 FORMAT(/21X,27HPPOOL SIZE VS TIME - MODEL T//)
22 FORMAT( 8X,4HTIME,8X,4HTIME,8X,4HSIZE,8X,4HSIZE,8X,4HSIZE)
23 FORMAT( 7X,6H(SECS),6X,6H(MINS),7X,5H(CMS),8X,3H(M),8X,4H(FT)//)
24 FORMAT( 5X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4)
25 FORMAT(1X,46H THE SPILL POOL IS CONFINED BY CHANNEL BANKS0 )
27 FORMAT(/17X,34H*** NOTE - AT APPROXIMATELY TIME =,G10.4,13H MINUTE
1S. ***/17X,45H*** THE POOL IS CONFINED BY CHANNEL BANKS ***)
31 FORMAT(1X,29H THE SPILL POOL IS CIRCULAR. )
35 FORMAT(5X,3H(M),9X,4H(FT),8X,6H(MINS),8X,5H(HRS),8X,5H(PPM),8X,5H(
1PPM)//)
51 FORMAT(/6X,35HTHIS CONCENTRATION IS EQUIVALENT TO,1X,G10.4,1X,7HPP
1M AND,1X,G10.4,1X,9HMG/LITER.)
52 FORMAT(/16X,48HTABLE OF PEAK CONCENTRATION VS TIME AND DISTANCE/
119X,41HAT MIDDEPTH AND BOTTOM OF RIVER - MODEL T//)
53 FORMAT(1X,10HX-DISTANCE,3X,10HX-DISTANCE,6X,4HTIME,9X,4HTIME,7X,
18HMID CONC,5X,8HBOI CONC)
54 FORMAT(1X,6(G10.4,3X))
55 FORMAT(/1X,55HNOTE - A CONCENTRATION IN PPM MULTIPLIED BY THE DEN
1SITY/1X,53HOF THE SPILLED CHEMICAL IN UNITS OF G/CM**3 GIVES THE/
21X,26HCONCENTRATION IN MG/LITER.)
56 FORMAT(/1X,64HTABLE OF CONCENTRATION VS TIME AT USER SPECIFIED PO
1INT - MODEL T//)
57 FORMAT(4X,4HTIME,11X,4HTIME,11X,4HCONC,11X,4HCONC)
58 FORMAT(3X,6H(MINS),10X,5H(HRS),10X,5H(PPM),7X,10H(MG/LITER)//)
59 FORMAT(1X,4(G10.4,5X))
61 FORMAT(/5X,35HTHE COORDINATES FOR THIS TABLE ARE-)
99 CALL TRACE(1,8,4)
END
SUBROUTINE FTCON(DW,WW,US,DIFW,CSAT,SPAMT,DENL,VISL,SURT,XN,T,X,Y,
1Z-DISRT,DISTM,PLCNX,CONC)

```

THIS SUBROUTINE CALCULATES THE DISSOLUTION RATE, TIME FOR ALL CHEMICAL TO DISSOLVE, DOWNSTREAM CONCENTRATION AT A USER SPECIFIED TIME AND LOCATION, AND THE DISTANCE DOWNSTREAM THE CENTER OF THE SPILL POOL HAS TRAVELED AT THE USER SPECIFIED TIME. THE ROUTINE IS PRIMARILY WRITTEN FOR A SPILL POOL WHOSE DIAMETER DOES NOT EXCEED THE WIDTH OF THE WATERBODY WHERE THE SPILL OCCURS, BUT IS APPROXIMATELY CORRECT FOR THE ALTERNATE CASE. IT IS MEANT ONLY TO BE USED UNDER STEADY-STATE FLOW CONDITIONS, I.E. A NON-TIDAL WATERBODY.

*****INPUTS*****

DW	= DEPTH OF RIVER	CMS
WW	= WIDTH OF RIVER	CMS
US	= MEAN STREAM VELOCITY	CM/SEC
DIFW	= DIFFUSION COEFFICIENT OF LIQUID IN WATER	CM2/SEC
CSAT	= SOLUBILITY OF CHEMICAL IN WATER	GM/GM
SPAMT	= AMOUNT OF CHEMICAL SPILLED	GM
DENL	= DENSITY OF CHEMICAL SPILLED	GM/CM3
VISL	= VISCOSITY OF CHEMICAL SPILLED	GM/CM-S
SURT	= SURFACE TENSION OF CHEMICAL SPILLED	D/CM
XN	= MANNING ROUGHNESS FACTOR FOR RIVER	NON-DIM
T	= TIME AT WHICH DOWNSTREAM CONCENTRATION DESIRED	SECS
X	= DOWNSTREAM DISTANCE FROM SPILL SITE AT WHICH CONCENTRATION DESIRED	CM

```

C      Y      = DISTANCE FROM MIDSTREAM AT WHICH CONCENTRATION
C      IS DESIRED
C      Z      = DEPTH IN RIVER AT WHICH CONCENTRATION DESIRED
C
C      *****OUTPUTS*****
C
C      DISRT= DISSOLUTION RATE OF CHEMICAL
C      DISTM= TIME FOR ALL CHEMICAL TO DISSOLVE
C      PLCNX= DISTANCE DOWNSTREAM POOL CENTER HAS TRAVELED BY
C      USER SPECIFIED TIME
C      CONC = CONCENTRATION OF CHEMICAL IN WATER AT SPECIFIED
C      TIME AND LOCATION
C
C      *****
C
C      DATA DENW,PI,IDIM,CHNLW,AREA/1.0,3.14159265,2,10000.,1.0/
C
C      CALCULATE DISSOLUTION RATE IN GM/CM2-SEC
C
C      VOL=SPAMT/DENL
C      SCLEN=0.1*DW
C      USQBR=0.1*US
C      BARK=1.46*SQRT(DIFW*USQBR/SCLEN)
C      DISRT=BARK*DENW*CSAT
C
C      FIND TIME AT WHICH ALL CHEMICAL HAS DISSOLVED
C
C      ISTOP=0
C      THICK=0.05
C      AREAC=SPAMT/(THICK*DENL)
C      DT=300.0
C      TIME=DT
C
C      40 CALL RLJSP(IDIM,VOL,DENL,VISL,SURT,TIME,CHNLW,SIZE)
C      AREA=PI*SIZE*SIZE
C      IF(AREA.GT.AREAC) AREA=AREAC
C      AREA=0.707*AREA
C      AMTDS=DISRT*AREA*TIME
C      IF(AMTDS-SPAMT) 25,20,20
C
C      25 IF(ISTOP.EQ.1) GO TO 26
C      TIME=2.*TIME
C      GO TO 40
C
C      20 ISTOP=1
C      TIME=TIME-120.0
C      IF(TIME.LE.0.0) TIME=60.
C      IF(TIME.EQ.60.) GO TO 26
C      GO TO 40
C
C      26 DISTM=TIME
C
C      CALCULATE CONCENTRATION IN WATER
C
C      RH=WW*DW/(2.*DW+WW)
C      USTAR=6.7305*XN*US/RH**(1./6.)
C      EZ=0.067*USTAR*RH
C      EX=0.1*EZ
C      IF(WW/DW-100.) 10,5,5
C
C      5 EY=0.1*EZ
C      GO TO 15
C
C      10 EY=0.23*USTAR*RH
C
C      15 SX=2.*SQRT(EX*T)
C      SY=2.*SQRT(EY*T)
C      SZ=2.*SQRT(EZ*T)
C      XBAR=X-(US*T)
C      YBAR=Y
C      ZBAR=Z
C      W=SQRT(AREA)
C      A=XBAR/SX
C      A=A*A
C      B=(YBAR/SY)**2.
C      C=((YBAR+W)/SY)**2.
C      D=(YBAR-W)/SY
C      D=D*D
C      E=(ZBAR/SZ)**2

```

```

F=((ZBAR-2.*DW)/SZ)**2
G=((ZBAR+2.*DW)/SZ)**2
CONC=CSAT*EXP(-A)*(EXP(-B)+EXP(-C)+EXP(-D))*(EXP(-E)+EXP(-F)+EXP(-
1G))
PLCNX=US*T
RETURN
END
SUBROUTINE TSPRD(DENL,DIA,HGT,FLOW,TIME,SIZE)

```

```

*****

```

THIS SUBROUTINE CALCULATES THE POOL RADIUS AS A FUNCTION OF TIME FOR AN INSOLUBLE, LIGHTER-THAN-WATER CHEMICAL WITH BOILING POINT GREATER THAN AMBIENT SPILLED CONTINUOUSLY ONTO WATER. IT WORKS ONLY FOR RADIAL SPREADING.

*** INPUTS ***

DENL	DENSITY OF SPILLED CHEMICAL	GM/CM3
DIA	AVERAGE DIAMETER OF HOLE	CM
HGT	HEIGHT OF HOLE ABOVE WATER	CM
FLOW	CONTINUOUS MASS RATE OF FLOW	GM/SEC
TIME	TIME AT WHICH POOL RADIUS DESIRED	SECS

*** OUTPUTS ***

SIZE	RADIUS OF POOL AT SPECIFIED TIME	CM
------	----------------------------------	----

```

*****

```

```

PI=3.141592654
G=980.
DENW=1.0
CALCULATING THE JET ENTRY PARAMETERS, HYDRAULIC JUMP RADIUS
AND THE RADIAL OUTFLOW VELOCITY.
FLW=FLOW/DENL
GRAV=G*(1.-DENL/DENW)
VEL=FLW/((PI/4.)*DIA*DIA)
U=SQRT((VEL**2.)+(2.*G*HGT))
A=DIA*SQRT(VEL/U)/2.
HB=A/2.
FB=2.*U*U/(GRAV*A)
FA=8.*FB/((SQRT((8.*FB)+1.))-1.)*3.
HA=HB*(FB/FA)**(1./3.)
UA=U*HB/HA
CALCULATING THE RADIAL SPREAD PARAMETERS.
TCH=A/UA
TAU=TIME/TCH
F=UA*UA/(HA*GRAV)
E1=0.41
E0=0.68
PSI1=SQRT(2./((1.-(2.*E1)))*SQRT(TAU-((F*E1)/((1.-E1-E1)**2.)))
CHI=PSI1+1.
SIZE=CHI*A
RETURN
END
OVERLAY(10,5)
PROGRAM MODV

```

SUBROUTINE MODV OBTAINS DATA FOR THE EXECUTION OF SUBROUTINE PKRHI, WHICH CALCULATES THE CONDITIONS AFTER WHOSE BOILING POINT IS GREATER THAN AMBIENT

```

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
DIMENSION AT(20),AV(20),AS(20),ATEM(20),AER(20),AR(20)
DIMENSION ASAV(20),ASAVT(20),PTITL1(6),PTITL2(6),PTITL3(6),
1PTITL4(6),PTITL5(6),XTITL(6),XTITL1(6),YTITL1(6),YTITL2(6),
1YTITL3(6),YTITL4(6),YTITL5(6)
EQUIVALENCE (XBX(1),AT(1)),(XRX(21),AV(1)),(XBX(41),AS(1)),
1(XBX(61),ATEM(1)),(XBX(81),AER(1)),(XBX(101),AR(1))
EQUIVALENCE (XBX(121),ASAV(1))

```

```

DATA MOD/4H V /
ODATA (PTITL1(I),I=1,6)/8HPOOL RAD,8HIUS/LENG,8HTH VS TI,
18HME - MOD,8HEL V ,1H /
ODATA (PTITL2(I),I=1,6)/8HVOLUME 0,8HF LIQUID,8H REMAINI,
18HNG VS TI,8HME - MOD,8HEL V /
ODATA (PTITL3(I),I=1,6)/8HEVAPORAT,8HION RATE,8H VS TIME,
18H - MODEL,8H V ,1H /
ODATA (PTITL4(I),I=1,6)/8HPOOL ARE,8HA VS TIM,8HE MODEL ,
18HV ,1H,1H /
ODATA (PTITL5(I),I=1,6)/8HTEMPERAT,8HURE OF L,8HIQUID VS,
18H TIME - ,8HMODEL V ,1H /
ODATA (XTITL (I),I=1,6)/8HELAPSED ,8HTIME FRO,8HM START ,
18HOF SPILL,8H.....(,8HMINUTES)/
ODATA (XTITL1(I),I=1,6)/8HELAPSED ,8HTIME FRO,8HM START ,
18HOF SPILL,8H.....,8H.(HOURS)/
ODATA (YTITL1(I),I=1,6)/8H RADIU,8HS/LENGTH,8H 0,
18HF POOL ,8H (,8HMETERS) /
ODATA (YTITL2(I),I=1,6)/1H ,8H VOLUME ,8H RE,
18HMAINING ,1H ,8H (M**3) /
ODATA (YTITL3(I),I=1,6)/8H EVA,8HPORATION,1H ,
18H RATE,8H (,8HKG/SEC) /
ODATA (YTITL4(I),I=1,6)/1H ,8H AREA ,1H ,
18HOF POOL ,1H ,8H (M**2) /
ODATA (YTITL5(I),I=1,6)/8H TEMP,8HERATURE ,8H OF,
18H LIQUID ,1H ,8H(DEG C) /
2 CONTINUE
CALL TRACE(0,8,5)
IR=0
LP=6
IS=6

```

C
C
C

OBTAIN DATA

```

CALL BEGPR(MOD)
CALL FRCL(1002,XMOL,IS,IR)
AM=XMOL
CALL FRCL(1004,DENL,IS,IR)
IF(DENL-1.0) 40,30,30
30 CALL PAGER(4)
WRITE(LP,100)
DENL=0.99
40 CONTINUE
CALL FRCL(1006,VISL,IS,IR)
IOUT=0
CALL FRCL(1007,CL,IS,IR)
CALL FRCL(1010,A,IS,IR)
CALL FRCL(1011,B,IS,IR)
CALL FRCL(1012,C,IS,IR)
CALL FRCL(1014,XLAT,IS,IR)
CALL FRCL(1021,DENLB,IS,IR)
CALL FRCL(1031,SURT,IS,IR)
CALL FRCL(2004,TINT,IS,IR)
CALL FRCL(2020,CHNLW,IS,IR)
CALL FRCL(2023,TW,IS,IR)
CALL FRCL(2033,CFIR,IS,IR)
CALL FRCL(2054,TA,IS,IR)
TMAX=TW
IF(TA.GT.TMAX) TMAX=TA
IF(TINT.GT.TMAX) TMAX=TINT
PRES=10.**(A-(B/(TMAX+C)))
CONC=100.*PRES/760.
CALL FRCL(2057,TIME,IS,IR)
IS1=6
CALL FRCL(4003,VOLI,IS1,IR)
C-----IF VOLUME IS NOT A DEFAULT, USE GIVEN VALUE
C OTHERWISE, COMPUTE USING MASS AND DENSITY
IF(IS1.GT.1) GO TO 300
CALL PAGER(1)
WRITE(LP,310)
CALL FRCL(4002,SPAMT,IS,IR)
VOLI=SPAMT/DENL
CALL FSU(4003,VOLI,4)

```

```

      IS1=4
300 IF(IS.GT.IS1) IS=IS1
      CALL PAGER(1)
      WRITE(LP,21)
      CALL COMPD(AM,TA,DENLB,DIFCO)
      CALL FSV(2053,DIFCO,4)
      CALL FRCL(2053,DIFCO,IS,IR)
      CALL IRCL(3010,IUPF,IS,IR)
      CALL IRCL(3014,ITAB,IS,IR)
      CALL EPRNT(MOD,IS,IR,IL)
      IF(IL.EQ.1) GO TO 99
      IF(IL.EQ.2) GO TO 2
      IOUT=0

C
C      CALL PKRHI
23 IDIM=2
      CALL PKRHI(IDIM,CHNLW,VOLI,DENL,VISL,SURT,XLAT,CL,A,B,C,TW,TIME,DI
1FCO,XNOL,VOL,SIZE,TEMP,SPEVA,THEND,AREA,IOUT)
      DIAM=2.*SIZE
      IF(DIAM.LE.CHNLW) GO TO 20
      IDIM=1
      CALL PKRHI(IDIM,CHNLW,VOLI,DENL,VISL,SURT,XLAT,CL,A,B,C,TW,TIME,DI
1FCO,XNOL,VOL,SIZE,TEMP,SPEVA,THEND,AREA,IOUT)
20 XCTIM=THEND+101.
      IF(TIME.LT.XCTIM) GO TO 22
      TIME=THEND+100.
      GO TO 23

C
C      UPDATE DATA BASE
22 CALL OUTPR(MOD)
      IF(IDIM.NE.2) GO TO 8
      CALL PAGER(1)
      WRITE(LP,11)
      8 IF(IDIM.NE.1) GO TO 9
      CALL PAGER(1)
      WRITE(LP,12)
      9 CALL ISV(2018,IDIM,4)
      CALL FSV(4026,VOL,4)
      CALL FSV(4027,SIZE,4)
      CALL FSV(4028,TEMP,4)
      CALL FSV(4029,SPEVA,4)
      CALL FSV(4030,THEND,4)
      CALL FSV(4031,AREA,4)
      AUTEM=(TINT+TEMP)/2.
      CALL FSV(4068,AUTEM,4)
      IF(CONC.LT.CFIR) CALL PAGER(3)
      IF(CONC.LT.CFIR) WRITE(LP,200)
      IF(CONC.LT.CFIR) CALL FSV(2033,0.0,6)

C
C      CALCULATE AND SAVE AVERAGE VAPOR EVOLUTION RATE UP TO USER
      SPECIFIED TIME OR OVER TIME IT TAKES POOL TO COMPLETELY VAPORIZE
      AND SAVE IT AND THE TIME IN CASE MODEL W FOLLOWS.

      TMX=TIME
      IF(THEND.LT.TMX) TMX=THEND
      IF(TMX.EQ.TIME) FLOW=(VOL1-VOL)*DENL/TMX
      IF(TMX.EQ.THEND) FLOW=VOL1*DENL/TMX
      IF(TMX.NE.TIME) GO TO 5
      CALL PAGER(3)
      WRITE(LP,13)
      5 IF(TMX.NE.THEND) GO TO 6
      CALL PAGER(3)
      WRITE(LP,14)
      6 CALL FSV(4044,FLOW,4)
      CALL FSV(4045,TMX,4)
      IF(TMX.LT.600.) CALL ISV(2061,0,4)
      IF(TMX.GE.600.) CALL ISV(2061,1,4)
      IDM=IDIM
      CALL ENDPR(MOD)

C
C      INTERROGATE USER PLOT REQUEST FLAG

```

```

C
C
C
C
C
      IF (IVPF.EQ.0.AND.ITAB.EQ.0) GO TO 99

      SET UP LOOP FOR CALCULATION OF PLOT ARRAYS OF VOLUME, SIZE,
      TEMPERATURE, SPECIFIC EVAPORATION RATE AND AREA OF SPILL
      VERSUS ELAPSED TIME

      DT=.95*TMEND/19.
      IDIM=2
      STM=0.0
      DO 10 I=1,20
      AT(I)=FLOAT(I-1)*DT+(TMEND/20.)
      IF(I.EQ.20) AT(20)=TMEND+100.0
      CALL PKRHI(IDIM,CHNLW,VOLI,DENL,VISL,SURT,XLAT,CL,A,B,C,TW,AT(I),
1      DIFCO,XMOL,AV(I),AS(I),ATEM(I),AER(I),TMNDE,AR(I),IOUT)
      DIAM=2.*AS(I)
      IF(DIAM.GT.CHNLW.AND.STM.EQ.0.0) STM=AT(I)/60.
      IF(STM.GT.0.0) IDIM=1
      AT(20)=TMEND
10 CONTINUE
      ATEM(20)=(2.*ATEM(19))-ATEM(18)
      AER(20)=(2.*AER(19))-AER(18)

C
C
      WRITE PLOT FILE
      IF (IVPF.EQ.0) GO TO 1
      DO 70 I=1,20
70 ASAV(I)=AT(I)/60.
      DO 71 I=1,5
      DO 72 II=1,20
      IF(I.EQ.1) ASAV(II)=AS(II)/100.
      IF(I.EQ.2) ASAV(II)=AV(II)/1000000.0
      IF(I.EQ.3) ASAV(II)=(AER(II)/1000.)*AR(II)
      IF(I.EQ.4) ASAV(II)=AR(II)/10000.
      IF(I.EQ.5) ASAV(II)=ATEM(II)
72 CONTINUE
      IF(I.EQ.1)
1CALL PLTLP(PTITL1,ASAV,ASAV,20,XTITL,YTITL1,1,60.,XTITL1)
      IF(I.EQ.2)
1CALL PLTLP(PTITL2,ASAV,ASAV,20,XTITL,YTITL2,1,60.,XTITL1)
      IF(I.EQ.3)
1CALL PLTLP(PTITL3,ASAV,ASAV,20,XTITL,YTITL3,1,60.,XTITL1)
      IF(I.EQ.4)
1CALL PLTLP(PTITL4,ASAV,ASAV,20,XTITL,YTITL4,1,60.,XTITL1)
      IF(I.EQ.5)
1CALL PLTLP(PTITL5,ASAV,ASAV,20,XTITL,YTITL5,1,60.,XTITL1)
      IF(I.NE.1.AND.IDIM.NE.1) GO TO 7
      CALL PAGER(4)
      IF(STM.GT.0.0) WRITE(LP,64) STM
7 IF(I.NE.4.AND.IDIM.NE.1) GO TO 71
      CALL PAGER(4)
      IF(STM.GT.0.0) WRITE(LP,64) STM
71 CONTINUE

C
C-----SET UP OFF-LINE PLOT
      PLTYP=1

C
C
C
C
      1 CONTINUE

      WRITE TABLE IF REQUESTED

      IF(ITAB.EQ.0) GO TO 99
      CALL PAGER(0)
      CALL PAGER(5)
      WRITE(LP,60)
      CALL PAGER(1)
      WRITE(LP,61)
      CALL PAGER(2)
      WRITE(LP,62)
      SSZ=0.0
      DO 50 I=1,20
      TMNS=AT(I)/60.
      SM=AS(I)/100.

```

```

SFT=AS(I)/(2.54*12.)
EVAP=AER(I)*AR(I)/1000.
AREA=AR(I)/10000.0
CALL PAGER(1)
WRITE(LP,63) TMNS,SM,SFT,ATEM(I),EVAP,AREA
50 CONTINUE
IF(STM.GT.0.0) CALL PAGER(4)
IF(STM.GT.0.0) WRITE(LP,64) STM
GO TO 99
11 FORMAT(1X,29H THE SPILL POOL IS CIRCULAR. )
12 FORMAT(1X,46H THE SPILL POOL IS CONFINED BY CHANNEL BANKS. )
13 FORMAT(/1X,64H THE VAPOR EVOLUTION RATE AT THE USER SPECIFIED TIME
1AND THE TIME/1X,34H ARE SAVED IN CASE MODEL W FOLLOWS.)
14 FORMAT(/1X,66H THE AVERAGE VAPOR EVOLUTION RATE WHILE THE POOL EVAP
1ORATES AND THE/1X,70H TIME FOR IT TO COMPLETELY EVAPORATE ARE SAVED
1 IN CASE MODEL W FOLLOWS.)
21 FORMAT(1X,56H THE DIFFUSION COEFFICIENT OF VAPOR IN AIR IS CALCULAT
1ED./1X,46H (VALUE IS DISPLAYED ON STORAGE, THEN RETRIEVAL /
2 2X,34H TO SHOW USER OVER-RIDE IF PRESENT))
310 FORMAT (1X,50H THE LIQUID VOLUME IS COMPUTED AS THE MASS/DENSITY.)
60 FORMAT(/36X,42H TABLE OF POOL CONDITIONS VS TIME - MODEL V//)
61 FORMAT(20X,4H TIME,10X,4H SIZE,10X,4H SIZE,10X,4H TEMP,8X,9HEVAP RATE,
17X,4H AREA)
62 FORMAT(19X,6H(MINS),10X,3H(M),10X,4H(FT),9X,7H(DEG C),7X,8H(KG/SEC
1),6X,6H(M**2)/)
63 FORMAT(19X,610.4,5(4X,610.4))
64 FORMAT(/17X,44H NOTE - DURING THE TIME INTERVAL BEFORE TIME=,G10.4
1,6H MINS,/17X,38H THE POOL IS CONFINED BY CHANNEL BANKS.)
100 FORMAT(96H WARNING- THE LIQUID DENSITY OF THE SPILLED CHEMICAL IS
*SO CLOSE TO WATER THAT IT MAY OR MAY NOT/77H FLOAT. FOR MODEL V IT
* WILL BE ASSUMED THAT IT HAS A DENSITY OF 0.99 GM/CM**3//)
200 FORMAT(72H NOTE-VAPOR PRESSURE OF LIQUID IS TOO LOW TO FORM FLAMMA
1BLE VAPOR CLOUD./ 53H LOWER FLAMMABLE LIMIT SET TO ZERO BEFORE PRO
2CEEDING./)
99 CALL TRACE(1,8,5)
END
SUBROUTINE FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,A)
C *** THIS SUBROUTINE IS CALLED BY THE RKGS SUBROUTINE FOR THE EVALUATIO
C *** OF THE RIGHT HAND SIDE OF THE SYSTEM OF THE DIFFERENTIAL EQUATIONS
C *** IN THIS CASE THE RHS FUNCTIONS ARE THE ONES THAT GIVE THE TEMPERAT
C *** DECREASE RATE AND THE MASS DECREASE RATE FOR THE HIGH VAPOR PRESSU
C *** LIQUID SPILL
C
DIMENSION Y(5),DERY(5)
PI=3.141592653
ETA=EXP(-BETA*(1./Y(1)-1./THETW))
TIME=X*TIMEC
C *** OBTAINING THE AREA OF THE POOL FROM SPREAD MODELS ***
CALL RLJSP(IDIM,VOLI,DENL,VISL,SURT,TIME,CHNLW,SIZE)
GO TO (5,10),IDIM
C *** A IS THE NON DIMENSIONAL AREA ---NON DIMENSIONLISED WITH RESPECT
C *** THE INITIAL CHARACTERISTIC AREA -AI--.
5 A=SIZE*CHNLW/AI
C *** C IS THE CORRECTION FACTOR WHICH TAKES INTO ACCOUNT THE NON UNIF
C *** THERMAL BOUNDARY LAYER THICKNESS IN WATER DURING THE SPREADING.
C=PI/2.
GO TO 15
10 A=PI*SIZE**2/AI
C=2.
C *** DERIVATIVE FUNCTIONS ***
15 IF(X)16,16,17
16 DERY(1)=-A*ETA
GO TO 18
17 DERY(1)=A*(DEL*C*(THETW-Y(1))/SQRT(X)-ETA)/Y(2)
18 DERY(2)=-A*ETA
RETURN
END
SUBROUTINE OUTP(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,AUX,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,A ,IHLF,NDIM,PMRT,IOUT,IHEAD)
C *** THIS IS AN OUTPUT SUBROUTINE CALLED BY THE RKGS SUBROUTINE.
C *** THIS ROUTINE IS USED HERE TO WRITE THE CALCULATED VALUES OF THE

```

```

C *** EVAPORATION RATE, MASS REMAINING ETC AT EVERY INSTANT OF TIME.
C *** X,Y, DERY ARE ALL IN NON DIMENSIONAL UNITS. THE OUTPUT VALUES ARE I
C *** DIMENSIONAL UNITS
C *****
C *** TIME AT WHICH THE ANSWERS ARE WRITTEN SECS
C *** TEMP TEMPERATURE OF THE LIQUID AT THE TIME INDICATED DEG C
C *** FRMAS FRACTION OF THE INITIAL MASS REMAINING IN THE SYSTEM
C *** ZMAS MASS REMAINING IN THE SYSTEM GMS
C *** AREA AREA OF THE POOL OF SPREAD AT THE TIME INDICATED CM
C *** SPEVA SPECIFIC EVAPORATION. THAT IS THE EVAPORATION IN GMS/SEC
C *** IOUT A FLAG WHICH INDICATES IF THE OUTP ROUTINE IS TO EXECUTED
C *** IHEAD A FLAG WHICH INDICATES IF THE HEADING IS TO BE WRITTEN OR
C *****
C DIMENSION AUX(8, 2), Y(5), DERY(5), PMRT(5)
C LP=6
C IF (AUX(5, 2) = 10, 10, 30) CHANGED TO CARD BELOW 04/24/73
C IF (Y(2)) 10, 30, 30
10 PMRT(5)=5.
20 RETURN
30 IF (IOUT) 20, 20, 40
40 IF (IHEAD) 70, 70, 60
60 CALL PAGER(3)
WRITE(LP, 200)
70 PI=3.141592653
TIME=X*TIMEC
TEMP=Y(1)*TEMPS-273.
FRMAS=Y(2)
ZMAS=ZMI*FRMAS
AREA=AXAI
SPEVA=0.
IF (A) 90, 90, 80
80 SPEVA=-ZMI*DERY(2)/(TIMEC*AREA)
90 CALL PAGER(1)
WRITE(LP, 100) TIME, FRMAS, ZMAS, SPEVA, AREA, IHLF
RETURN
100 FORMAT(2X, E14.5, 2X, 2(E14.5, 4X), 2X, E14.5, 8X, E14.5, 4X, I2)
200 FORMAT(6X, 4HTIME, 12X, 11HFRAC TN MASS, 6X, 14HREMAINING MASS, 6X, 14HSPE
1CIFIC EVAPN, 9X, 9HPOOL AREA/6X, 4HSECS, 33X, 5HGRAMS, 13X, 12HGM/SEC-CM*
2*2, 13X, 5HSQ CM//)
END
SUBROUTINE PKRHI(IDIM, CHNLW, VOLI, DENL, VISL, SURT, XLAT, CL, A, B, C, TW,
1TIME, DIFCO, XMOL, VOL, SIZE, TEMP, SPEVA, TMEND, AREA, IOUT)
C *****
C *** THIS SUBROUTINE CALCULATES THE CONDITIONS AFTER THE SPILL OF A HIGH
C *** VAPOR PRESSURE LIQUID ON WATER. FOR A GIVEN SPILL VOLUME AND TEM
C *** THE SUBROUTINE RETURNS THE SIZE OF SPREAD, THE VOLUME REMAINING,
C *** AND THE EVAPORATION RATE OF THE LIQUID. THE CLAUSIUS CLAYPERON
C *** EQUATION FOR SATURATED VAPOR PRESSURE IS UTILIZED.
C *****
C ***** A R G U M E N T D E S C R I P T I O N *****
C ***** INPUT VALUES *****
C IDIM DIMENSION OF THE SPREAD 1-DIMENSIONAL=1 RADIAL=2
C CHNLW CHANNEL WIDTH CM
C VOLI INITIAL VOLUME CUBIC CM
C DENL DENSITY OF THE SPILLED LIQUID G/CC
C VISL VISCOSITY OF THE SPILLED LIQUID DYNE-SEC/CM
C SURT SURFACE TENSION OF THE SPILLED LIQUID DYNES/CM
C XLAT HEAT OF VAPORIZATION OF THE SPILLED LIQUID CAL/GM
C CL SPECIFIC HEAT OF THE SPILLED LIQUID CAL/GM-DEGREE K
C A, B, C CONSTANTS IN THE SATURATED VAPOR PRESSURE EQUATION
C P=10.**((A-(B/(T+C))))
C TW TEMPERATURE OF WATER DEGREES C
C TIME TIME AT WHICH THE CONDITIONS ARE TO BE KNOWN SEC
C DIFCO DIFFUSION COEFFICIENT OF VAPOR IN AIR SQ CM/SEC
C XMOL MOLECULAR WEIGHT OF SPILLED COMPOUND GM/MOL
C *****
C ***** OUTPUT VALUES *****
C TMEND TIME AT WHICH PROGRAM STOPS - SPECIFIED TIME OR TIME AT WH
C ALL LIQUID HAS EVAPORATED - WHICHEVER COMES FIRST SEC
C VOL VOLUME OF THE LIQUID REMAINING AT TMEND
C SIZE SIZE OF THE SPILL AT TMEND (RADIUS OR LENGTH) CMS

```

```

C      SPEVA   SPECIFIC EVAPORATION RATE AT TMEND          GM/SEC C
C      AREA    AREA OF THE SPILL AT TMEND                SQUARE CM
C ***** INPUT/OUTPUT VALUES *****
C      IOUT    1 IF WANT INTERMEDIATE VALUES OF VOL, SIZE, TEMP
C              0 IF ONLY WANT FINAL VALUES
C              VALUE MAY BE CHANGED WITHIN SUBROUTINE CALLED BY THIS ONE
C *****
C      EXTERNAL OUTP, FCT
C      DIMENSION PMRT(5),Y(5),AUX(8,2),DERY(5)
C      FACT=10.
C      NSEC=50
C      CALL      HMTG(DIFCO,XMOL,VOLI,HMP)
C      DENW=1.0
C      CONDW=.0013
C      PI=3.141592653
C *** CALCULATION OF CHARACTERISTIC AND NON DIMENSIONALING PARAMETERS
C      ALFW=CONDW/DENW
C      ZMI=VOLI*DENL
C      AI=FACT*VOLI*.666667
C      TEMPS=XLAT/CL
C      PVAPI=(10.**(A-(B/(TW+C))))/760.
C      EIDOT=HMP*PVAPI
C      TIMEC=ZMI/(AI*EIDOT)
C      TOWEN=TIME/TIMEC
C      DEL=CONDW*TEMPS/(EIDOT*XLAT*SQRT(PI*ALFW*TIMEC))
C      BETA=B/TEMPS
C      THETW=(TW+273.)/TEMPS
C *** INTEGRATION OF THE SIMULTANEOUS SYSTEM OF DIFFERENTIAL EQUATIONS
C *** Y(1) IS THE TEMPERATURE AND Y(2) IS THE MASS OF THE LIQUID ***
C *** FIRST THE INITIAL CONDITIONS AND ERROR WEIGHTS ARE GIVEN ***
2000 Y(1)=THETW
C      Y(2)=1.
C      NDIM=2
C      PMRT(1)=0.
C      PMRT(2)=TOWEN
C      PMRT(3)=TOWEN/FLOAT(NSEC)
C      PMRT(4)=.005
C      DERY(1)=0.5
C      DERY(2)=0.5
C      CALL      PKRRK(PMRT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX,AI,ZMI,TEMPS,
C      1 TIMEC,DEL,BETA,THETW,IDIM,CHNLW,VOLI,DENL,VISL,SURT,SIZE,S,IOUT,X)
C      FRMAS=AUX(1,2)
C      ZMAS=ZMI*FRMAS
C      VOL=ZMAS/DENL
C      TEMP=AUX(1,1)*TEMPS-273.
C      AREA=S*AI
C      SPEVA=0.
C      IF(S) 90,90,80
80   SPEVA=-ZMI*AUX(2,2)/(TIMEC*AREA)
90   TMEND=X*TIMEC
C      RETURN
C      END
C      SUBROUTINE PKRRK(PMRT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX,AI,ZMI,TEMPS,
C      1 TIMEC,DEL,BETA,THETW,IDIM,CHNLW,VOLI,DENL,VISL,SURT,SIZE,S,IOUT,X)
C      EXTERNAL FCT,OUTP
C      DIMENSION Y(5),DERY(5),AUX(8,2),A(4),B(4),C(4),PMRT(5)
C      DO 1 I=1,NDIM
1    AUX(8,I)=.06666667*DERY(I)
C      X=PMRT(1)
C      XEND=PMRT(2)
C      H=PMRT(3)
C      PMRT(5)=0.
C      CALL      FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,
C      1 CHNLW,VOLI,DENL,VISL,SURT,SIZE,S)
C      ERROR TEST
C      IF(H*(XEND-X))38,37,2
C      PREPARATIONS FOR RUNGE-KUTTA METHOD
2    A(1)=.5
C      A(2)=.2928932

```

```

A(3)=1.707107
A(4)=.1666667
B(1)=2.
B(2)=1.
B(3)=1.
B(4)=2.
C(1)=.5
C(2)=.2928932
C(3)=1.707107
C(4)=.5
C PREPARATIONS OF FIRST RUNGE-KUTTA STEP
DO 3 I=1,NDIM
  AUX(1,I)=Y(I)
  AUX(2,I)=DERY(I)
  AUX(3,I)=0.
3 AUX(4,I)=0.
  IREC=0
  H=H+H
  IHLF=-1
  ISTEP=0
  IEND=0
C RECORDING OF INITIAL VALUES OF THIS STEP
  IHEAD=1
  CALL      OUTP(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,AUX,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S      ,IREC,NDIM,PRMT,IOUT,IHEAD)
  IHEAD=0
C START OF A RUNGE-KUTTA STEP
4 IF((X+H-XEND)*H)7,6,5
5 H=XEND-X
6 IEND=1
7 CONTINUE
  IF(PRMT(5))40,8,40
8 ITEST=0
9 ISTEP=ISTEP+1
C START OF INNERMOST RUNGE-KUTTA LOOP
  J=1
10 AJ=A(J)
  BJ=B(J)
  CJ=C(J)
  DO 11 I=1,NDIM
    R1=H*DERY(I)
    R2=AJ*(R1-BJ*AUX(4,I))
    Y(I)=Y(I)+R2
    R2=R2+R2+R2
11 AUX(4,I)=AUX(4,I)+R2-CJ*R1
  IF(J-4)12,15,15
12 J=J+1
  IF(J-3)13,14,13
13 X=X+.5*H
14 CALL      FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S)
  GOTO 10
C END OF INNERMOST RUNGE-KUTTA LOOP
C TEST OF ACCURACY
15 IF(ITEST)16,16,20
  IN CASE ITEST=0 THERE IS NO POSSIBILITY FOR TESTING OF ACCURACY
16 DO 17 I=1,NDIM
17 AUX(4,I)=Y(I)
  ITEST=1
  ISTEP=ISTEP+ISTEP-2
18 IHLF=IHLF+1
  X=X-H
  H=.5*H
  DO 19 I=1,NDIM
    Y(I)=AUX(1,I)
    DERY(I)=AUX(2,I)
19 AUX(4,I)=AUX(3,I)
  GOTO 9
C IN CASE ITEST=1 TESTING OF ACCURACY IS POSSIBLE
20 IMOD=ISTEP/2
  IF(ISTEP-IMOD-IMOD)21,23,21
21 CALL      FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,

```

```

1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S)
DO 22 I=1,NDIM
AUX(5,I)=Y(I)
22 AUX(7,I)=DERY(I)
GOTO 9
C COMPUTATION OF TEST VALUE DELT
23 DELT=0.
DO 24 I=1,NDIM
24 DELT=DELT+AUX(8,I)*ABS(AUX(4,I)-Y(I))
IF(DELT-PRMT(4))28,28,25
C ERROR IS TOO GREAT
25 IF(IHLF-10)26,36,36
26 DO 27 I=1,NDIM
27 AUX(4,I)=AUX(5,I)
ISTEP=ISTEP+ISTEP-4
X=X-H
IEND=0
GOTO 18
C RESULT VALUES ARE GOOD
28 CALL FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S)
CALL OUTP(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,AUX,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S,IREC,NDIM,PRMT,IOUT,IHEAD)
C *** CHECK IS MADE IF THE NEXT MASS VALUE IS NEGATIVE. IF SO CALCULATI
C *** IS STOPPED BY GOINT TO LINE 50.
ISKIP=PRMT(5)/2.
IF(ISKIP-2) 55,50,55
55 DO 29 I=1,NDIM
AUX(1,I)=Y(I)
AUX(2,I)=DERY(I)
AUX(3,I)=AUX(6,I)
Y(I)=AUX(5,I)
29 DERY(I)=AUX(7,I)
IF(PRMT(5))40,30,40
30 DO 31 I=1,NDIM
Y(I)=AUX(1,I)
31 DERY(I)=AUX(2,I)
IREC=IHLF
IF(IEND)32,32,39
C INCREMENT GETS DOUBLED
32 IHLF=IHLF-1
ISTEP=ISTEP/2
H=H+H
IF(IHLF)4,33,33
33 IMOD=ISTEP/2
IF(ISTEP-IMOD-IMOD)4,34,4
34 IF(DELT-.02*PRMT(4))35,35,4
35 IHLF=IHLF-1
ISTEP=ISTEP/2
H=H+H
GOTO 4
C RETURNS TO CALLING PROGRAM
36 IHLF=11
CALL FCT(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,DEL,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S)
GOTO 39
37 IHLF=12
GOTO 39
38 IHLF=13
39 CALL OUTP(X,Y,DERY,AI,ZMI,TEMPS,TIMEC,AUX,BETA,THETW,IDIM,
1CHNLW,VOLI,DENL,VISL,SURT,SIZE,S,IREC,NDIM,PRMT,IOUT,IHEAD)
40 RETURN
C *** DIFM AND DIFT ARE THE AVERAGE VALUES OF THE RATE OF CHANGE OF MAS
C *** RATE OF CHANGE OF TEMPERATURE, RESPECTIVELY BETWEEN THE LAST STEP
C *** AND THE TIME AT WHICH THE MASS GOES NEGATIVE.
50 DIFM=(AUX(2,2)+DERY(2))/2.
DIFT=(AUX(2,1)+DERY(1))/2.
C *** DX IS THE TIME BETWEEN THE LAST TIME AND THE TIME AT WHICH THE MA
C *** GOES TO ZERO
DX=(0.-AUX(1,2))/DIFM
X=X+DX
AUX(1,2)=0.

```

```

AUX(1,1)=AUX(1,1)+DIFT*DX
RETURN
END
OVERLAY(10,6)
PROGRAM MODX

```

PROGRAM EXECUTES MODEL X, INDEX 24

MODX CALCULATES THE CONDITIONS FOLLOWING A SPILL OF A HEAVIER-THAN-WATER, SLIGHTLY SOLUBLE CHEMICAL WITH A BOILING POINT GREATER THAN THE AMBIENT TEMPERATURE. SEE ROUTINES SINK1, SINK2, AND SINK3 FOR SPECIFIC CAPABILITIES.

```

COMMON/C/PLTYP,XBX(150)
INTEGER PLTYP
EQUIVALENCE (XBX(1),XD(1))
EQUIVALENCE (XBX(21),CNC(1,1))
EQUIVALENCE (XBX(61),CRTM(1,1))
DIMENSION XD(20),CNC(2,20),CRTM(2,20)
DATA MOD/4H X /
1 CONTINUE

CALL TRACE(0,8,6)
LP=6
IS=6
IR=0

OBTAIN DATA

CALL BEGPR(MOD)
CALL FRCL(1002,AM,IS,IR)
CALL FRCL(1003,TBOIL,IS,IR)
CALL FRCL(1004,DENL,IS,IR)
IF(DENL - 1.) 10,10,20
10 CALL PAGER(6)
WRITE(6,100)
DENL=1.01
20 CONTINUE
CALL FRCL(1021,DLB,IS,IR)
CALL FRCL(1025,TCRIT,IS,IR)
CALL FRCL(1031,SURT,IS,IR)
CALL FRCL(2021,DS,IS,IR)
CALL FRCL(2023,TWAT,IS,IR)
CALL FRCL(2039,X,IS,IR)
CALL FRCL(2040,Y,IS,IR)
CALL FRCL(2041,Z,IS,IR)
CALL FRCL(2044,DW,IS,IR)
CALL FRCL(2045,WW,IS,IR)
CALL FRCL(2047,US,IS,IR)
CALL FRCL(2052,XN,IS,IR)
CALL FRCL(4002,SPAHT,IS,IR)

CALL ROUTINE SOLUB TO FIND SOLUBILITY OF CHEMICAL AT THE
WATER TEMPERATURE
SOLUB CALLS DATA OF FIELD NUMBERS 1026,1028, AND 1029

CALL SOLUB(TWAT,CSAT,IS,IR)
CSAT=CSAT/100.

CALCULATE DIFFUSION COEFF OF CHEMICAL IN WATER

CALL CDIFW(AM,DLB,TWAT,TCRIT,TBOIL,DIFW)
CALL FSV(2043,DIFW,4)
CALL FRCL(2043,DIFW,IS,IR)

CALL IRCL(3012,ITX,IS,IR)
IF(ITX.GE.1) CALL FRCL(2031,XX,IS,IR)
CALL EPRNT(MOD,IS,IR,IL)
IF(IL.EQ.1) GO TO 99
IF(IL.EQ.2) GO TO 1

```

```

C
C      CALL SINK
      CALL SINK1(SURT,DW,US,SPAMT,DENL,XN,DS,TIME,DIST,AREA,PLEN,TMSPR)
      CALL SINK2(DIFW,US,PLEN,AREA,CSAT,SPAMT,DW,DISRT,DISTH)
      CALL SINK3(WW,DW,X,Y,Z,US,DIST,TIME,DISRT,DISTH,XN,AREA,PLEN,CONC,
1CLRTH,IFLAG)
C
C      UPDATE DATA BASE
      CALL OUTPR(MOD)
      CALL FSV(4032,TIME,4)
      CALL FSV(4033,DIST,4)
      CALL FSV(4036,DISRT,4)
      CALL FSV(4037,DISTH,4)
      CALL FSV(4038,AREA,4)
      CALL FSV(4039,PLEN,4)
      CALL FSV(4040,TMSPR,4)
      CALL FSV(4041,CLRTH,4)
      IF(CONC.GT.DENL) CONC=DENL
      CALL FSV(4042,CONC,4)
      CMGL=CONC*1000000.0
      CPPM=CMGL/DENL
      CALL PAGER(2)
      WRITE(LP,52) CPPM,CMGL
      IF(IFLAG.NE.1) GO TO 75
      CALL PAGER(4)
      WRITE(6,50)
      CALL PAGER(1)
      WRITE(LP,51)
      CALL FSV(2039,X,6)
75  CONTINUE
      CALL ENDPR(MOD)
      IF(ITX.EQ.0) GO TO 99
C
C      CALCULATE DATA FOR MAX CONCENTRATIONS AT RIVER BOTTOM AND SURFACE
      VS DISTANCE DOWNSTREAM
      CALL PAGER(0)
      CALL PAGER(9)
      WRITE(6,30)
      WRITE(6,31)
      WRITE(6,40)
      WRITE(6,41)
      Y=0.0
      Z=0.0
      XMN=DIST+PLEN+(10.*2.54*12.)
C
C      SET MAX DIST TO 5 TIMES MIN ALLOWED IF USER SUPPLIES UNREASONABLE
      INPUT
      IF(XMX.LE.XMN) XMX=5.*XMN
      DX=(XMX-XMN)/19.
      DO 90 II=1,2
      DO 80 I=1,20
      JJJJ=I
      XD(I)=XMN+((FLOAT(JJJJ)-1.)*DX)
      CALL SINK3(WW,DW,XD(I),Y,Z,US,DIST,TIME,DISRT,DISTH,XN,AREA,PLEN,
1CNC(II,I),CRTH(II,I),IFLAG)
      IF(CNC(II,I).GT.DENL) CNC(II,I)=DENL
      XDIS=XD(I)/(2.54*12.)
      XD(I)=XD(I)/100.
      CONC=CNC(II,I)*1000000.
      CNC(II,I)=CONC/DENL
      TMS=CRTH(II,I)/60.
      THR=CRTH(II,I)/3600.
      CALL PAGER(1)
      WRITE(LP,70) XD(I),XDIS,CONC,CNC(II,I),TMS,THR
80  CONTINUE
      IF(II.EQ.2) GO TO 99
      CALL PAGER(0)
      CALL PAGER(9)
      WRITE(6,30)

```

```

WRITE(6,32)
WRITE(6,40)
WRITE(6,41)
Z=DW
90 CONTINUE
GO TO 99
30 FORMAT (/12X,42HTABLE OF MAXIMUM CONCENTRATION VS DISTANCE)
31 FORMAT(20X,26HAT WATER SURFACE - MODEL X//)
32 FORMAT(20X,25HAT WATER BOTTOM - MODEL X//)
40 FORMAT(3X,8HDISTANCE,4X,8HDISTANCE,6X,4HCONC,8X,4HCONC,6X,8HCLR TI
1ME,4X,8HCLR TIME)
41 FORMAT(6X,3H(M),8X,4H(FT),5X,10H(MG/LITER),5X,5H(PPM),6X,6H(MINS),
17X,5H(HRS)//)
50 FORMAT(/40H NOTE-USER INPUT FOR X WAS UNREASONABLE./ 1X,43HIT WAS
*CHANGED AS NOTED IN THE HACS MANUAL./)
51 FORMAT(1X,19HTHE NEW X VALUE IS-)
52 FORMAT(/1X,35HTHIS CONCENTRATION IS EQUIVALENT TO,G10.4,1X,7HPPM A
1ND,G10.4,1X,9HMG/LITER.)
70 FORMAT(2X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4,2X,G10.4)
100 FORMAT(/70H WARNING - THE CHEMICAL WHICH HAS BEEN DISCHARGED HAS A
* LIQUID DENSITY/68H SO CLOSE TO WATER THAT IT MAY OR MAY NOT SINK.
* FOR MODEL X, IT WILL/50H BE ASSUMED THAT IT HAS A DENSITY OF 1.01
* GM/CM**//)
99 CONTINUE
CALL TRACE(1,8,6)
END
SUBROUTINE SINK1(SURT,DW,US,SPAMT,DENL,XN,DS,TIME,DIST,AREA,PLEN,
*TMSPR)

```

C*****

THIS SUBROUTINE CALCULATES THE TIME TO SINK, DISTANCE TRAVELED
DOWNSTREAM BEFORE REACHING RIVER BOTTOM, AREA OF RIVERBED COVERED,
LENGTH OF CHEMICAL POOL ON RIVERBED, AND TIME FOR POOL TO SPREAD
TO ITS MAXIMUM EXTENT FOR A HEAVIER-THAN-WATER, SLIGHTLY SOLUBLE
CHEMICAL WITH A BOILING POINT GREATER THAN THE AMBIENT
TEMPERATURE SPILLED INTO A NON-TIDAL RIVER.

*****INPUTS*****

SURT =	INTERFACIAL TENSION OF SPILLED CHEMICAL	DYNE/CM
DW =	DEPTH OF RIVER	CM
US =	MEAN STREAM VELOCITY	CM/SEC
SPAMT =	AMOUNT OF CHEMICAL SPILLED	GRAMS
DENL =	DENSITY OF LIQUID SPILLED	GM/CM3
XN =	MANNING ROUGHNESS FACTOR FOR RIVER	NON-DIM
DS =	DISTANCE FROM SURFACE WHERE RELEASE TAKES PLACE	
	DS=0.0 AT SURFACE, DS=DW AT BOTTOM	CM

*****OUTPUTS*****

TIME =	TIME FOR CHEMICAL TO SINK TO RIVERBED	SECS
DIST =	DISTANCE CHEMICAL TRAVELS DOWNSTREAM BEFORE REACHING RIVERBED	CM
AREA =	AREA OF RIVERBED COVERED BY CHEMICAL	CM**2
PLEN =	LENGTH OF CHEMICAL POOL ON RIVERBED	CM
TMSPR =	TIME FOR POOL TO SPREAD TO ITS MAXIMUM EXTENT	SECS

*****OTHER PARAMETERS*****

WC =	CRITICAL WEBER NUMBER AT WHICH THE LIQUID BREAKS UP. (8)WC(10)	
CD =	DRAW COEFFICIENT DURING THE DECENT OF THE DROP IN WATER.	
GRAV =	EFFECTIVE GRAVITY (G*(DENL/DENW-1.))	CM/SEC2
GR =	GRAVITATIONAL ACCELERATION	CM/SEC2
PRW =	PRANDTT NUMBER FOR WATER.	

C*****

```

PI=3.141592654
A=1.778
GR=980.

```

১৯৮৬

1

הנה

2

הצגה

[illegible]

הנהלה

מה זה?

מבוא

CC

```

C      DISTM= TIME FOR ALL CHEMICAL TO DISSOLVE                      SECS
C
C      ***** OTHER PARAMETERS *****
C      S      = SHAPE FACTOR-DEPENDS ON THE SHAPE OF THE LIQUID
C                POOL-MEAN VALUE OF SEVERAL STANDARD GEOMETRICAL
C                SHAPES=1.076
C      SC      = SCHMIDT NUMBER=KINEMATIC VISCOSITY OF WATER/
C                DIFFUSIVITY OF LIQUID IN WATER
C      REL      = REYNOLDS NUMBER FOR THE STREAM FLOW BASED ON
C                POOL LENGTH, AND MEAN STREAM VELOCITY.
C*****
C      DATA DENW,VISKW,S/1.0,0.01,1.076/
C
C      CALCULATE DISSOLUTION RATE OF LIQUID POOL
C
C      SC=VISKW/DIFW
C      REL=US*PLEN/VISKW
C      W=AREA/PLEN
C      DISRT=S*0.0343*DENW*CSAT*DIFW*W*(SC**(11./27.))*(REL**(7./9.))
C      1*((PLEN/DW)**(1./9.))
C      DISTM=SPAMT/DISRT
C      RETURN
C      END
C      SUBROUTINE SINK3(WW,DW,X,Y,Z,US,DIST,TIME,DISRT,DISTM,XN,AREA,PLEN
C      1,CONC,CLRTM,IFLAG)
C*****
C      THIS SUBROUTINE GIVES THE DOWN STREAM CONCENTRATION
C      OF A CHEMICAL DISSOLVING SLOWLY INTO THE WATER
C      FROM A LIQUID POOL ON THE RIVER BED.  THE DISPERSION
C      CALCULATIONS ARE PERFORMED BY ASSUMING A LINE
C      SOURCE AT THE DOWNSTREAM EDGE OF THE LIQUID POOL.
C
C      *****INPUTS*****
C
C      WW      = WIDTH OF RIVER                      CMS
C      DW      = DEPTH OF RIVER                      CMS
C      X      = DOWNSTREAM DISTANCE FROM SPILL SITE AT WHICH
C                CHEMICAL CONCENTRATION IS DESIRED    CMS
C      Y      = DISTANCE FROM MIDSTREAM WHERE CONCENTRATION DESIRED CMS
C      Z      = DEPTH IN RIVER WHERE CONCENTRATION IS DESIRED CMS
C      US      = MEAN STREAM VELOCITY                CM/SEC
C      DIST    = DISTANCE CHEMICAL TRAVELS DOWNSTREAM BEFORE
C                REACHING RIVERBED                    CMS
C      TIME    = TIME FOR CHEMICAL TO SINK TO RIVERBED SECS
C      DISRT   = AVERAGE DISSOLUTION RATE OF CHEMICAL POOL GM/SEC
C      DISTM   = TIME FOR ALL CHEMICAL TO DISSOLVE    SECS
C      XN      = MANNING ROUGHNESS FACTOR FOR RIVER  NON-DIM
C      AREA    = AREA OF RIVERBED COVERED BY CHEMICAL CM**2
C      PLEN    = LENGTH OF CHEMICAL POOL ON RIVERBED  CM
C
C      *****OUTPUTS*****
C
C      CONC    = CONCENTRATION OF CHEMICAL AT DOWNSTREAM
C                POINT SPECIFIED                      GM/CM3
C      CLRTM   = TIME AFTER SPILL AT WHICH POLLUTANTS
C                WILL HAVE PASSED DOWNSTREAM POINT SPECIFIED SECS
C      IFLAG   = FLAG INDICATING WHETHER X DISTANCE INPUT WAS
C                RETURNED MODIFIED OR UNCHANGED. IFLAG=0 MEANS
C                IT WAS UNCHANGED, IFLAG=1 MEANS THE DISTANCE
C                INPUT WAS BEHIND OR WITHIN THE SPILL POOL AND
C                THE DISTANCE WAS CHANGED TO X=DIST+PLEN+X  NON-DIM
C*****
C      CALCULATE DISPERSION COEFFICIENTS FOR NON TIDAL RIVER
C
C      PI=3.141592654
C      DENW=1.0
C      W=AREA/PLEN

```

```

      RH=WW*DW/(2.*DW+WW)
      USTAR=6.7305*XN*US/RH**(1./6.)
      EZ=0.067*USTAR*RH
      EX=0.1*EZ
      IF(WW/DW-100.) 10,5,5
5     EY=0.1*EZ
      GO TO 15
10    EY=0.23*USTAR*RH
15    XD=X-(DIST+PLEN)
      IF(XD.LE.0.0) IFLAG=1
      IF(IFLAG.EQ.1) XD=X
      IF(IFLAG.EQ.1) X=XD+DIST+PLEN

      ABOVE MANIPULATION PREVENTS UNREASONABLE X DISTANCE INPUT

      DETERMINE WHETHER NEAR- OR FAR-FIELD CONCENTRATION MODEL IS USED

      XCRIT=((WW/W)**2.)*(US*DW/EZ)*DW
      IF(XD-XCRIT) 20,20,25

      CALCULATE WATER CONCENTRATION USING NEAR-STREAM MODEL
20    SZ=SQRT(4.)*(D*EZ/US)
      SY=SQRT(4.)*(D*EY/US)
      DOTM=DISRT/W
      CMAX=DOTM/(2.*SQRT(PI*US*XD*EZ))
      B=W/2.
      ESP=ERF((Y-WW+B)/SY)-ERF((Y-WW-B)/SY)+ERF((Y+WW+B)/SY)-ERF((Y+WW-B
1) /SY)
      A=((DW-Z)/SZ)**2
      AA=((Z+DW)/SZ)**2
      C1=CMAX*(EXP(-A)+EXP(-AA))
      C2=ERF((Y+B)/SY)-ERF((Y-B)/SY)+ESP
      CONC=C1*C2
      GO TO 30

      CALCULATE WATER CONCENTRATION USING FAR-STREAM MODEL
25    DOTM=DISRT/W
      CONC=DISRT/(WW*DW*US)

      CALCULATE TIME AT WHICH ALL CHEMICAL GOES BY SPECIFIED POINT XYZ
30    CLRTM=TIME+DISTM+(XD/US)
      RETURN
      END
-END OF FILE-

```

6. CYBERNET USE

Given the HACS/UIM source program file, and the external data files used by HACS, the execution of HACS is obtained by issuing the appropriate commands after logging onto the Cybernet system. Authorization and access procedures for use are provided by the National Response Center.

Execution of HACS requires the process of compiling the source program file(s), loading these programs together with system utilities and library routines, then executing the resulting overlay load module file. On Cybernet, these procedures are carried out as separate steps (during development) to prepare the load module file, and the instructions to access this file are stored in a separate file (procedure). These instructions are:

```
ATTACH,UIMABS.  
ATTACH,TAPE9=PCKPRP.  
GET,TAPE10=HACST10.  
ATTACH,TAPE11=FLDTXT.  
ATTACH,TAPE12=SCNTXT.  
GET,TAPE13=MODDIR.  
UIMABS.
```

These instructions are stored in a procedure file named UIMRUN, and the only user entry required (after log-on) is to type -UIMRUN.

The procedure establishes the required linkages between the internal HACS file references and the external files cataloged on Cybernet. The current external files accessed on Cybernet are:

```
PCKPRP = chemical properties file  
HACST10 = default file  
FLDTXT = data item explanatory messages  
SCNTXT = scenario descriptions  
MODDIR = model descriptions
```

The file UIMABS contains the HACS/UIM program code load modules in overlay form. Files UIMABS, PCKPRP, FLDTXT, and SCNTXT are direct access files; HACST10 and MODDIR are indirect access files.

The statements ATTACH and GET also establish correspondence (e.g., via TAPE9=PCKPRP) from the internal HACS/UIM file reference numbers 9, 10, 11, 12 and 13 to the appropriate external files.

7. ASSOCIATED PROGRAMS

During the development of the HACS/UIM, a number of associated programs were developed and are briefly described in this section together with their listings. These programs are grouped into four classifications:

- 7.1 Chemical Property File Manipulation,
- 7.2 Message File Creation,
- 7.3 Message File Display, and
- 7.4 Utilities.

The property file manipulation programs include the conversion from the prior fixed length format used on both Cybernet and the CDC 3300 to the variable length format containing both model and scenario codes used on Cybernet with the UIM. Additional programs relating directly to UIM use in this section include property file data editing and conversions between variable length format and a variable logical record length format packed into a fixed physical record length format. Finally, this section includes three additional programs used to obtain analyses of property file content: an index of chemical recognition codes by hazard assessment model; an index of chemical recognition codes by hazard assessment path codes; and, data gap identification.

Sections 7.2 and 7.3 contain the programs used to build, then display, respectively, the message text files for data fields, scenarios and assessment models. The master programs for file building and display functions were first created for the field text file, then modified as necessary for the scenario and model messages. It was considered likely that over time changes or refinements in explanations for the data fields would be desired, and these programs include capabilities for file maintenance. The creation of the scenario and model descriptions was considered less susceptible to change (and they are also smaller), so file maintenance capabilities were not included.

Two additional programs are included in Section 7.4 as utilities. The first is a very generalized set of individual, and related, data coding and uncoding utility routines which were developed for use in compacting file structures such as the HACS physical property file. These routines are very generalized, and only a sub-set is used within HACS and related property file programs. The second program in Section 7.4 is a straight-forward utility used during terminal entry of program code and message text to provide a translation of tab key entries.

7.1 Chemical Property File Manipulation

7.1.1 Model and Scenario Coding

Program CMPRS, listed on the following pages, is used to create a modified version of the property file in which model and scenario codes are appended to each chemical data record, provision for missing chemical data items in the record is deleted, and a file of variable length records is created.

The program requires the use of the data compression utility routines given in Section 7.4.1.

A fixed record length version of the property file is read on input, and the file header (elements unchanged) is copied to output. Next a loop is entered to read each individual chemical data record. The original model codes in the hazard assessment path code from the file are compared to a list of code letters in the program, and a code is set for each model in the sequence determined by the internal list. The resulting codes are then packed for output into a single word. Note that this translation produces a completely uniform specification of hazard assessment model codes for all 900 chemicals.

After a verification of the model letter coding, the model letter codes established from the input file are then used with specific logic in the program to define all hazard assessment scenario codes applicable for the particular chemical. These codes are created and then packed into a single code word for output with the chemical record. A verification step (uncoding and compare) is also included to test the validity of the coded output.

Finally a third coding procedure is performed to omit all chemical data items which are missing (formerly stored as a value of 0.0). The chemical data item status codes (0 = missing, 2 = estimate, 3 = exact) are packed into a five word array for output, and the spaces formerly allocated for missing items are then deleted. The final coded variable length chemical data record is then written to the output file using the statement BUFFER OUT (to obtain the desired variable length characteristic on Cybernet).

The program produces a printed audit of the conversion/compaction process at the user terminal.

81/09/18. 09.31.28.
PROGRAM SAVPCK

```

PROGRAM CMPRS(OUTPUT,TAPE6=OUTPUT,TAPE9,TAPE10)
DIMENSION FREF(84),IPTH(16),MFLG(29),YNAM(5),YPTH(8),YVAL(74)
0 INTEGER BUFF(17),FCM,FLN,FND,FNE,FNM,FNX,FSN,HDR(6),OREC(84),
1 OTP,PTLST(30),SCLST(28),SCOD,SF01,SF02,SF03,SF04,SF05,
2 SF06,SF07,SF08,SF09,SF10,SF11,SF12,SF13,SF14,SF15,SF16,
3 SF17,SF18,SF19,SF20,SF21,SF22,SF23,SF24,SF25,SF26,SF27,
4 SF28,SFLG(28),TCOD(5),YCOD,YTYP(74)
0 EQUIVALENCE (FREF(1),OREC(1)),(HDR(1),OREC(1)),(IBLNK,PTLST(30)),
1 (MCOD,OREC(7)),(SCOD,OREC(8)),(TCOD(1),OREC(9)),
2 (YCOD,YVAL(1)),(YNAM(1),OREC(2))
0 EQUIVALENCE (MSWB,MFLG(2)),(MSWC,MFLG(3)),(MSWD,MFLG(4)),
1 (MSWE,MFLG(5)),(MSWF,MFLG(6)),(MSWH,MFLG(8)),
2 (MSWI,MFLG(9)),(MSWK,MFLG(11)),(MSWL,MFLG(12)),
3 (MSWM,MFLG(13)),(MSWO,MFLG(15)),(MSWP,MFLG(16)),
4 (MSWQ,MFLG(17)),(MSWR,MFLG(18)),(MSWT,MFLG(20)),
5 (MSWU,MFLG(21)),(MSWV,MFLG(22)),(MSWX,MFLG(24)),
6 (MSWY,MFLG(25)),(MSWZ,MFLG(26)),(MSWII,MFLG(27)),
7 (MSWRR,MFLG(28)),(MSWSS,MFLG(29))
0 EQUIVALENCE (SF01,SFLG(1)),(SF02,SFLG(2)),(SF03,SFLG(3)),
1 (SF04,SFLG(4)),(SF05,SFLG(5)),(SF06,SFLG(6)),
2 (SF07,SFLG(7)),(SF08,SFLG(8)),(SF09,SFLG(9)),
3 (SF10,SFLG(10)),(SF11,SFLG(11)),(SF12,SFLG(12)),
4 (SF13,SFLG(13)),(SF14,SFLG(14)),(SF15,SFLG(15)),
5 (SF16,SFLG(16)),(SF17,SFLG(17)),(SF18,SFLG(18)),
6 (SF19,SFLG(19)),(SF20,SFLG(20)),(SF21,SFLG(21)),
7 (SF22,SFLG(22)),(SF23,SFLG(23)),(SF24,SFLG(24)),
8 (SF25,SFLG(25)),(SF26,SFLG(26)),(SF27,SFLG(27)),
9 (SF28,SFLG(28))

C
DATA ITP/9/,LP/6/,OTP/10/
0 DATA (PTLST(I),I=1,30)/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,
1 1HK,1HL,1HM,1HN,1HO,1HP,1HQ,1HR,1HS,1HT,1HU,1HV,1HW,1HX,1HY,
2 1HZ,2HII,2HRR,2HSS,1H /
0 DATA (SCLST(I),I=1,28)/3HA B,3HA C,5HA B C,5HA D E,7HA D F G,
1 9HA D E F G,3HA H,5HA I J,7HA H I J,5HA K L,7HA K M N,
2 9HA K L N N,3HA O,3HA P,5HA P Q,7HA P R S,9HA P Q R S,
3 3HA T,5HA T U,5HA U W,9HA T U V W,3HA X,5HA X Y,1HZ,2HII,
4 2HRR,4HRR C,2HSS/

C
C-----REWIND INPUT PROPERTY FILE AND READ HEADER RECORD. TERMINATE
C IF GET END OF FILE.
C REWIND ITP
C READ(ITP) HDR
C IF(EOF(ITP)) 220,10

C
C-----REWIND OUTPUT FILE AND WRITE VARIABLE LENGTH HEADER RECORD.
C NOTE THAT RECORD IS WRITTEN IN ODD PARITY AND STATUS TEST IS
C REQUIRED BEFORE FURTHER EXECUTION.
10 REWIND OTP
C BUFFER OUT(OTP,1) (HDR(1),HDR(6))
C IF(UNIT(OTP)) 20,230,240

C
C-----INITIALIZE SUMMARY TOTALS FOR FILE PROCESSING COUNTS OF
C CONVERTED ITEMS AND WRITE OUTPUT HEADER
20 FCM=0
FLN=0
FNX=0
FNE=0
FND=0
FNM=0
FSN=0
WRITE(LP,990)

C
C-----RETURN HERE TO READ NEXT PHYSICAL PROPERTY RECORD
30 READ(ITP) YTYP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)
C

```

```

C-----TEST FOR END OF FILE ON INPUT.  WRITE EOF TO OUTPUT, REWIND
C      BOTH TAPES, WRITE FILE CONVERSION SUMMARY TOTALS AND STOP.
      IF(EOF(ITP)) 40,50
40  ENDFILE OTP
      REWIND OTP
      REWIND ITP
      WRITE(LP,1000) FCM,FLN,FNX,FNE,FND,FNM,FSN
      STOP

C-----CHEMICAL NAME READ ON INPUT IS EQUIVALENCED TO DESIRED
C      LOCATION IN OUTPUT BUFFER.  PREPARE TO CODE MODEL LETTERS
C      BY FIRST UNPACKING PATH CODES IN A8 FORMAT TO A4 FORMAT.
C      INCREMENT COUNT OF CHEMICAL RECORDS AND MOVE CHEMICAL
C      RECOGNITION CODE TO OUTPUT BUFFER.
50  FCM=FCM+1
      OREC(1)=YCOD
      DECODE(80,1010,YPTH) IPTH

C-----LOOP ON ALL NON-BLANK MODEL CODES IN ARRAY IPTH, LOOK EACH UP
C      IN MASTER LIST PTLST.  IF MATCH TO ELEMENT PTLST(I), SET
C      INDICATOR MFLG(I) TO 1, OTHERWISE LEAVE INITIAL VALUE MFLG(I)
C      OF ZERO UNCHANGED.  WRITE ERROR MESSAGE IF ANY NON-BLANK
C      UNRECOGNIZABLE MODEL CODES ARE FOUND ON INPUT.
      DO 60 I=1,29
60  MFLG(I)=0
      DO 80 I=1,16
          ITHP=IPTH(I)
          IF(ITHP.EQ.IBLNK) GO TO 80
          DO 70 J=1,29
              IF(ITHP.NE.PTLST(J)) GO TO 70
              MFLG(J)=1
              GO TO 80
70  CONTINUE
          WRITE(LP,1030) ITHP
          STOP
80  CONTINUE

C-----SETTINGS OF ARRAY ELEMENT MFLG(I) ARE NOW 0 IF MODEL I NOT
C      PRESENT, 1 IF PRESENT.  PACK INTO SINGLE CODE WORD MCOD FOR
C      OUTPUT.
      CALL INIT(MCOD,29,1,1)
      CALL PACK(MFLG,MCOD)

C-----READ EACH SETTING OF PACKED CODE WORD MCOD, COMPARE TO
C      MFLG FOR VERIFICATION AND MOVE UP TO 10 PATH CODE LETTERS
C      INTO PRINT BUFFER FOR DISPLAY.
      K=0
      DO 90 I=1,29
          ITHP=ITST(MCOD,I)
          IF(ITHP.EQ.MFLG(I)) GO TO 85
          WRITE(LP,1050) I
          STOP
85  IF(ITHP.EQ.0) GO TO 90
          IF(K.GE.10) GO TO 90
          K=K+1
          BUFF(K)=PTLST(I)
90  CONTINUE
92  IF(K.GE.10) GO TO 94
          K=K+1
          BUFF(K)=IBLNK
          GO TO 92

C-----USE SETTINGS IN ARRAY MFLG TO IDENTIFY SCENARIOS.  SET VALUES
C      IN ARRAY SFLG TO 1 IF SCENARIO I IS PRESENT, 0 OTHERWISE.
C      START WITH MODELS RR AND C FOR SCENARIOS RR, RR-C.
94  DO 100 I=1,28
100 SFLG(I)=0
      IF(MSWRR.EQ.0) GO TO 120
      IF(MSWC.EQ.0) GO TO 110
      SF27=1
      GO TO 170
110 SF26=1

```

```

C
C-----SCENARIOS FOR MODELS D, Z, II, SS
120 SF13=MSWQ
    SF24=MSWZ
    SF25=MSWII
    SF28=MSWSS

C
C-----SCENARIOS A-B, A-C AND A-B-C
    SF01=MSWB
    SF02=MSWC
    SF03=(MSWB+MSWC)/2

C
C-----SCENARIOS A-D-E, A-D-F-G AND A-D-E-F-G
    SF04=MSWE
    SF05=MSWF
    SF06=(MSWE+MSWF)/2

C
C-----SCENARIOS A-H, A-I-J AND A-H-I-J
    SF07=MSWH
    SF08=MSWI
    SF09=(MSWH+MSWI)/2

C
C-----SCENARIOS A-K-L, A-K-M-N AND A-K-L-M-N
    SF10=MSWL
    SF11=MSWM
    SF12=(MSWL+MSWM)/2

C
C-----SCENARIOS A-P, A-P-Q, A-P-R-S AND A-P-Q-R-S
    IF(MSWP.EQ.0) GO TO 130
    SF15=MSWQ
    SF16=MSWR
    ITMP=MSWQ+MSWR
    SF17=ITMP/2
    IF(ITMP.EQ.0) SF14=1

C
C-----SCENARIOS A-T, A-T-U, A-V-W AND A-T-U-V-W. IF MODELS U AND V
C      ARE GIVEN WITHOUT T, ONLY SCENARIO A-V-W IS OBTAINED. IF T
C      AND V ARE GIVEN WITHOUT U, ONLY A-T IS OBTAINED SINCE A-T-V-W
C      IS INVALID AS A COMPLETE SCENARIO.
130 IF(MSWT.EQ.0) GO TO 150
    IF(MSWU.EQ.1) GO TO 140
    SF18=1
    GO TO 160
140 SF19=1
    SF21=MSWU
150 SF20=MSWU

C
C-----SCENARIOS A-X AND A-X-Y
160 IF(MSWX.EQ.0) GO TO 170
    SF23=MSWY
    IF(MSWY.EQ.0) SF22=1

C
C-----SETTINGS OF ARRAY ELEMENT SFLG(I) NOW DEFINE PRESENCE (1) OR
C      ABSENCE (0) OF SCENARIO I. PACK INTO SINGLE CODE WORD SCOD
C      FOR OUTPUT.
170 CALL INIT(SCOD,28,1,1)
    CALL PACK(SFLG,SCOD)

C
C-----READ EACH SETTING OF PACKED CODE WORD SCOD AND COMPARE TO
C      SFLG FOR VERIFICATION. MOVE SCENARIO CODE LETTERS INTO PRINT
C      BUFFER FOR DISPLAY. NOTE THAT A LIMIT OF 7 SCENARIOS CAN BE
C      DISPLAYED.
DO 180 I=1,28
    ITMP=ITST(SCOD,I)
    IF(ITMP.EQ.SFLG(I)) GO TO 175
    WRITE(LP,1070) I
    STOP
175 IF(ITMP.EQ.0) GO TO 180
    FSN=FSN+1
    IF(K.GE.17) GO TO 180
    K=K+1
    BUFF(K)=SCLST(I)

```

AD-A110 275

LITTLE (ARTHUR D) INC CAMBRIDGE MA
DEVELOPMENT OF A HACS USER INTERFACE MODULE.(U)
SEP 81 R G POTTS

F/6 9/2

UNCLASSIFIED

USCG-D-74-81

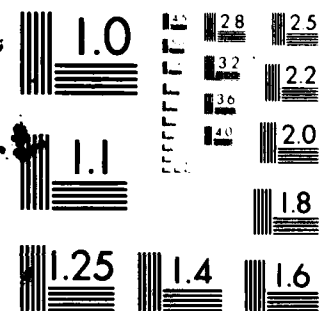
DOT-CG-925331-A
NL

3 13

01/01/85



END
DATE
FILMED
2 82
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```

180 CONTINUE
C-----CONVERT CHEMICAL DATA ITEM STATUS CODES READ ON INPUT AS ONE
C CODE PER ELEMENT OF ARRAY YTP TO PACKED FORMAT IN CODE WORD
C ARRAY TCD. NOTE THAT CODE VALUES ALLOWED ARE 0,1,2,3 WHICH
C PROVIDES FOR ADDITION OF CHEMICAL SPECIFIC DEFAULT ITEMS AS
C CODE 1. THE CODE ARRAY TCD CONTAINS 5 WORDS OF WHICH 30 BITS
C EACH ARE USED. THE PACKED CODES ARE EACH 2 BITS IN LENGTH SO
C A TOTAL OF 75 CODES CAN BE STORED. SINCE ONLY 74 CODES ARE
C REQUIRED, PACKING IS DONE BY SUBROUTINE SET (SUBROUTINE PACK
C WOULD REQUIRE ALL 75 CODES TO BE STORED).
      CALL INIT(TCD,30,5,2)
      DO 190 I=1,74
190 CALL SET(TCD,I,YTP(I))
C-----READ EACH SETTING OF PACKED DATA ITEM STATUS CODES AND
C COMPARE TO ORIGINAL VALUE IN YTP ARRAY FOR VERIFICATION.
C USE PACKED CODES TO COUNT DATA ITEM TYPES.
      NX=0
      NE=0
      ND=0
      NM=0
      DO 200 I=1,74
      ITMP=ITST(TCD,I)
      IF(ITMP.EQ.YTP(I)) GO TO 195
      WRITE(LP,1080) I
      STOP
195 IF(ITMP.EQ.0) NM=NM+1
      IF(ITMP.EQ.1) ND=ND+1
      IF(ITMP.EQ.2) NE=NE+1
      IF(ITMP.EQ.3) NX=NX+1
200 CONTINUE
      FNM=FNM+NM
      FND=FND+ND
      FNE=FNE+NE
      FNX=FNX+NX
C-----MOVE ALL NON-MISSING DATA ITEM VALUES IN ARRAY YVAL TO ARRAY
C FREF FOR OUTPUT. CODING ASSUMES ONE-WORD FORMAT FOR BOTH
C INTEGERS AND REAL VALUES, SO THAT FIRST DATA VALUE IS MOVED
C INTO POSITION 14 OF OUTPUT RECORD.
      LN=13
      DO 210 I=4,74
      IF(YTP(I).EQ.0) GO TO 210
      LN=LN+1
      FREF(LN)=YVAL(I)
210 CONTINUE
      FLN=FLN+LN
C-----DISPLAY SUMMARY OUTPUT FOR COMPACTED RECORD
      WRITE(LP,1090) YCD,LN,NX,NE,ND,NM,(BUFF(I),I=1,K)
C-----WRITE COMPRESSED DATA RECORD TO OUTPUT FILE, THEN RETURN
C FOR NEXT CHEMICAL AFTER ERROR FREE OUTPUT.
      BUFFER OUT(OTP,1) (OREC(1),OREC(LN))
      IF(UNIT(OTP)) 30,230,240
C-----NON-RECOVERABLE ERROR CONDITIONS
220 WRITE(LP,1100)
      STOP
230 WRITE(LP,1110)
      STOP
240 WRITE(LP,1120)
      STOP
C
9900FORMAT (//75H SUMMARY REPORT ON SCENARIO CODING AND COMPACTION OF
1CHEMICAL PROPERTY FILE//44H COD LN NX NE ND NM HAZARD ASSESSMENT M
2ODELS,7X,76HSCENARIO 1 SCENARIO 2 SCENARIO 3 SCENARIO 4 SCENARIO 5
3 SCENARIO 6 SCENARIO 7/20H --- -- -- -- -- -- ,30(1H-),7(1X,10(1H-
4)))
10000FORMAT (//5X,22HCONVERTED FILE TOTALS://5X,15,33H VARIABLE LENGTH
1CHEMICAL RECORDS/5X,15,33H WORDS FOR COMPACTED FILE STORAGE/5X,15,

```

```

225H EXACT DATA VALUES STORED/5X,15,29H ESTIMATED DATA VALUES STORE
3D/5X,15,27H DEFAULT DATA VALUES STORED/5X,15,28H MISSING DATA VALU
4ES SKIPPED/5X,15,22H SCENARIO CODES STORED///)
1010 FORMAT (8(2A4,2X))
1030 FORMAT (5X,26H*****UNKNOWN MODEL CODE = ,A10)
1050 FORMAT (5X,25H*****ERROR ON MODEL CODE ,I2,13H VERIFICATION)
1070 FORMAT (5X,28H*****ERROR ON SCENARIO CODE ,I2,13H VERIFICATION)
1080 FORMAT (5X,26H*****ERROR ON STATUS CODE ,I2,13H VERIFICATION)
1090 FORMAT (1X,A3,5(1X,I2),1X,10A3,7(1X,A10))
1100 FORMAT (5X,44H*****ERROR - HEADER ON INPUT PRECEDED BY EOF)
1110 FORMAT (5X,21H*****EOF ERROR ON OTP)
1120 FORMAT (5X,24H*****PARITY ERROR ON OTP)
      END
READY.

```

7.1.2 Variable to Fixed Length Conversion

On Cybernet, it was found that variable length records were written in either a variable length or fixed length format depending on the program statements used, and the operating procedures used (e.g., PACK). The BUFFER OUT/BUFFER IN statements currently used in HACS/UIM provide, without packing the chemical data file, the desired variable length structure.

A brief review of the standard Fortran features available on both DEC and PRIME computers indicated that some difficulty in building similar variable length file structures might be encountered in conversion, and two additional programs were created.

The first, PROGRAM CONV, is listed in the following section, and provides for the conversion of a variable record length file created by PROGRAM CMPRS (refer to Section 7.1.1) to a file in which the variable length records are packed into fixed length physical records. Output is performed by unformatted WRITE statements instead of BUFFER OUT.

Subroutine MVLRC (refer to Section 7.1.3) provides for input conversion of the file created by this program. Neither CONV nor MVLRC are used by the current version of HACS/UIM on Cybernet; these are provided for future use on DEC or PRIME installations as may be desired.

81/09/18. 09.33.47.
PROGRAM RGPBTB

PROGRAM CONV(OUTPUT,TAPE6=OUTPUT,TAPE9,TAPE10)

PROGRAM CONV (FOR CONVERT) READS AN INPUT CHEMICAL PROPERTY FILE CONTAINING VARIABLE LENGTH RECORDS CREATED BY USING THE BUFFER OUT STATEMENT, AND A RECORD LENGTH IS APPENDED TO THE BEGINNING OF EACH RECORD. UNFORMATTED, FIXED LENGTH BINARY RECORDS ARE WRITTEN TO THE OUTPUT FILE. THE LAST OUTPUT FIXED LENGTH RECORD IS FILLED WITH ZEROES IF NECESSARY. FOR SUBSEQUENT SEQUENTIAL INPUT FROM THE FIXED LENGTH RECORD FILE, THE LOGICAL RECORD LENGTH IN THE RECORD IS REQUIRED TO CONTROL INPUT OF FIXED LENGTH RECORDS TO RECONSTRUCT THE DESIRED VARIABLE LENGTH LOGICAL RECORD.

BLEN = FIXED LENGTH OF OUTPUT RECORD BUFFER, SET HERE TO 63
FOR CORRESPONDENCE WITH MINIMUM CDC MASS STORAGE
DISK ALLOCATION
I = INDEX TO ELEMENTS OF VARIABLE LENGTH INPUT RECORD
IREC = BUFFER FOR STORAGE OF INPUT VARIABLE LENGTH RECORD.
MAXIMUM RECORD LENGTH (ON INPUT) IS 84 WORDS TO
TO BE STORED IN POSITIONS 2 TO 85 OF IREC.
ITP = I/O UNIT NUMBER FOR INPUT FILE
J = INDEX TO ELEMENTS IN FIXED LENGTH OUTPUT BUFFER,
CYCLES BETWEEN 1 AND BLEN BY 1.
LEN = ADJUSTED LENGTH OF VARIABLE LENGTH INPUT RECORD (ONE
IS ADDED TO ADJUST FOR STORAGE OF LENGTH COUNT)
LP = I/O UNIT NUMBER FOR PARITY ERROR MESSAGE
OREC = OUTPUT BUFFER, FIXED LENGTH = BLEN
OTP = I/O UNIT NUMBER FOR OUTPUT FILE

COMMON VARIABLES USED - NONE

SUBROUTINES REQUIRED - LENGTH,UNIT

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/318A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 16 JANUARY 1981

INTEGER BLEN,OREC,OTP
DIMENSION IREC(85),OREC(63)
EQUIVALENCE (LEN,IREC(1))
DATA BLEN/63/,ITP/9/,LP/6/,OTP/10/

-----INITIALIZE, REWIND BOTH FILES AND SET OUTPUT BUFFER POINTER
TO EMPTY VALUE.

REWIND ITP
REWIND OTP
J=0

-----RETURN HERE TO READ EACH NEW INPUT RECORD UNTIL EOF OR PARITY
ERROR IS ENCOUNTERED. INPUT RECORD IS READ INTO ARRAY IREC
FROM POSITION 2 ON. TEST FOR EOF OR PARITY ERROR AND BRANCH
TO PROCESS. OTHERWISE OBTAIN LENGTH OF INPUT RECORD, AND
STORE LENGTH OF OUTPUT RECORD (=INPUT LENGTH + 1) IN POSITION
1 OF IREC.

10 BUFFER IN(ITP,1) (IREC(2),IREC(85))
IF(UNIT(ITP)) 20,40,60
20 LEN=LENGTH(ITP)+1

-----MOVE WORDS 1 TO LEN FROM INPUT ARRAY IREC TO POSITION IN OUTPUT
ARRAY OREC. EACH TIME OREC IS FILLED TO CAPACITY (BLEN), WRITE
OUTPUT RECORD AND RE-INITIALIZE TO STORE NEXT WORD FROM IREC

```

C      IN FIRST POSITION OF OREC.
      DO 30 I=1,LEN
      J=J+1
      IF(J.LE.BLEN) GO TO 30
      WRITE(OTP) OREC
      J=1
      30 OREC(J)=IREC(I)
C
C-----AFTER ALL ELEMENTS OF IREC HAVE BEEN MOVED TO OUTPUT, RETURN
C      TO CONTINUE CYCLE FOR NEXT INPUT RECORD UNTIL EOF ON INPUT IS
C      REACHED.
      GO TO 10
C
C-----EOF ON INPUT UNIT HAS BEEN REACHED. FILL LAST OUTPUT BUFFER
C      WITH ZEROES, THEN WRITE OREC AND TERMINATE RUN.
      40 J=J+1
      IF(J.GT.BLEN) GO TO 50
      OREC(J)=0
      GO TO 40
      50 WRITE(OTP) OREC
      ENDFILE OTP
      GO TO 70
C
C-----PARITY ERROR CONDITION. WRITE ERROR MESSAGE AND TERMINATE RUN.
      60 WRITE(LP,1000)
      70 REWIND OTP
      REWIND ITP
      STOP
      1000 FORMAT (47H *****PARITY ERROR ON INPUT TAPE. JOB ABORTED.//)
      END
READY.

```

7.1.3 Fixed to Variable Length Conversion

Referring to Section 7.1.2 for additional information, Subroutine MVLRC listed on the following pages can be used to read a blocked fixed length file of chemical property data and extract the desired variable length chemical data record. Program TEST which precedes MVLRC in the listing is a short test program used to read the data file, and print selected information. This program illustrates the method used to read the data file, and can be used as a model for HACS/UIM revision if the file structure is changed during conversion. Subroutine MVLRC does not currently appear in the Cybernet operating version of HACS/UIM. Note that subroutine MVLRC processes the input file using only unformatted READ statements.

81/09/18. 09.36.15.
PROGRAM RGPMLR

```

PROGRAM TEST(OUTPUT,TAPE6=OUTPUT,TAPE9)
COMMON/SAVE/IERR,IREF(85),J
INTEGER YCOD
EQUIVALENCE (LEN,IREF(1)),(YCOD,IREF(2))
K=0
10 CALL MVLRC(K)
WRITE(6,1000) K,IERR,J,LEN,YCOD
IF(IERR.EQ.0) GO TO 10
STOP
1000 FORMAT (3X,I3,3X,I1,3X,I2,3X,I2,3X,A3)
END
SUBROUTINE MVLRC(K)

```

SUBROUTINE MVLRC (FOR MOVE LOGICAL RECORD) READS A VARIABLE LENGTH LOGICAL RECORD FROM AN EXTERNAL FILE OF UNFORMATTED (BINARY) FIXED LENGTH RECORDS CONTAINING RECORD LENGTH POINTERS. THIS ROUTINE IS INTENDED TO BE USED TO MOVE THE LOGICAL RECORDS SEQUENTIALLY FROM THE BEGINNING OF THE FILE TO THE END OF THE FILE. EACH CALL RETURNS THE NEXT LOGICAL RECORD, STORED IN THE COMMON BLOCK SAVE. PRIOR TO THE FIRST USE, THE CALLING PROGRAM SHOULD SET THE ARGUMENT K TO ZERO.

BLEN = FIXED LENGTH OF EXTERNAL FILE RECORDS, SET TO 63 FOR CORRESPONDENCE WITH PROGRAM USED TO CREATE THE FILE.
 I = INDEX TO ELEMENTS IN VARIABLE LENGTH RECORD TO BE RETURNED TO CALLING PROGRAM
 IERR = END OF FILE STATUS INDICATOR RETURNED TO CALLING PROGRAM. 0 IS NORMAL RETURN, 1 INDICATES END OF FILE. ON RETURN WITH IERR=1, THE CONTENTS OF IREF ARE NOT CHANGED, K HOWEVER IS UPDATED.
 IREF = AREA FOR STORAGE OF VARIABLE LENGTH DATA RECORD, UP TO A MAXIMUM OF 85 WORDS IN LENGTH. LENGTH, LEN, IS STORED IN POSITION 1 ON RETURN IF IERR=0.
 J = INDEX TO ELEMENTS IN FIXED LENGTH BUFFER, CYCLES BETWEEN 1 AND BLEN BY 1. NOTE THAT J IS SAVED IN COMMON AND INITIALIZATION IS FORCED BY INITIAL CALL WITH K=0.
 K = NUMERICAL INDEX TO VARIABLE LENGTH RECORDS. ON INPUT, VALUE GIVES SEQUENTIAL NUMBER OF RECORD RETURNED BY LAST CALL. ON OUTPUT, K GIVES NUMBER OF RECORD RETURNED BY CURRENT CALL.
 LEN = TOTAL LENGTH OF VARIABLE LENGTH RECORD, STORED ON RETURN IN FIRST LOCATION OF IREF. MAY BE ZERO ON RETURN FOR END OF FILE.
 OREF = FIXED LENGTH RECORD BUFFER, LENGTH = BLEN
 OTP = I/O UNIT NUMBER FOR FIXED LENGTH RECORD FILE

COMMON VARIABLES USED - IREF,IREF,J,LEN

SUBROUTINES REQUIRED - EOF

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
 35/318A ACORN PARK,
 CAMBRIDGE, MASS., 02140
 TEL. 617-864-5770 EXT. 2813

DATE - 16 JANUARY 1981

```

COMMON/SAVE/IERR,IREF(85),J
EQUIVALENCE (LEN,IREF(1))

INTEGER BLEN,OREF(63),OTP
DATA BLEN/63/,OTP/9/

```

```

C-----FOR INITIAL CALL WITH K=0, REWIND FIXED LENGTH RECORD FILE AND
C      SET BUFFER POINTER TO FORCE FIRST READ.
C      IF(K.NE.0) GO TO 10
C      REWIND OTP
C      J=BLEN

C-----INITIALIZE FOR EACH NEW RECORD TO BE RETURNED. SET ERROR FLAG
C      IERR TO NO ERROR VALUE, AND MOVE K TO NUMBER OF RECORD TO BE
C      RETURNED. INDEX J IS INCREMENTED TO FIRST POSITION OF NEXT
C      LOGICAL RECORD WHICH CONTAINS THE LENGTH OF THAT RECORD.
C      10 IERR=0
C         K=K+1
C         J=J+1

C-----IF POINTER J EXCEEDS LENGTH OF FIXED LENGTH BUFFER, READ NEXT
C      BUFFER AND RESET POINTER TO FIRST POSITION. JUMP TO EOF RETURN
C      IF END OF FILE IS FOUND, OTHERWISE SAVE LENGTH OF NEXT LOGICAL
C      RECORD. NOTE THAT ZERO FILL IN LAST FIXED LENGTH BUFFER ALSO
C      INDICATES END OF FILE CONDITION.
C      IF(J.LE.BLEN) GO TO 30
C      READ(OTP) OREC
C      IF(EOF(OTP)) 60,20
C      20 J=1
C      30 LEN=OREC(J)
C      IF(LEN.EQ.0) GO TO 60

C-----MOVE ELEMENTS 2 TO LEN INTO VARIABLE LENGTH RECORD FIELDS.
C      READ NEW BUFFER EACH TIME POINTER J MOVES OUT OF RANGE OF
C      FIXED LENGTH BUFFER. NOTE EOF TEST IS USED FOR EACH READ, AND
C      THAT THIS CODE ASSUMES LEN .GE. 2. RETURN WHEN DONE.
C      DO 50 I=2,LEN
C         J=J+1
C         IF(J.LE.BLEN) GO TO 50
C         READ(OTP) OREC
C         IF(EOF(OTP)) 60,40
C         40 J=1
C         50 IREC(I)=OREC(J)
C         RETURN

C-----END OF FILE RETURN. SET IERR AND RETURN.
C      60 IERR=1
C         RETURN
C         END
READY.

```

7.1.4 CDC 3300 Conversion

Program CONV which is listed on the following pages has evolved as the primary means of transferring the chemical property data from the CDC 3300 to other computers. On the CDC 3300 the selective retrieval and display program is run with options selected to output all non-missing data items for all chemicals to an output tape. The tape contains one data item per output record, in the format required for property file updates. (Refer to documentation on the separate retrieval and update programs for additional detailed information.)

Program CONV reads the tape file written on the CDC 3300, and produces as output on the host computer a file written in binary format for use with HACS. Functions performed by CONV include setting status codes, inserting missing data values (0.0) which are not present in the input file, and collecting all input records for a single chemical into a single output record.

The output file produced is formatted as a series of fixed length binary records (original HACS property file format).

81/09/18. 11.49.46.
PROGRAM RGPCNV

```

PROGRAM CONV(OUTPUT,TAPE6=OUTPUT,TAPE9,TAPE10)
INTEGER EST,HDR,OTP,XCOD,YTYP
REAL FBLNK,YNAM,YPTH,YVAL
INTEGER IBLNK
INTEGER I,IRUF,ITAG,ITP,NFLD,NWCO
DIMENSION HDR(6),IBUF(7),YNAM(5),YPTH(8),YTYP(74),YVAL(74)
EQUIVALENCE (IBLNK,FBLNK),(NWCO,YVAL(1))
DATA EST/1HE/,ITP/9/,OTP/10/,IBLNK/1H /
DATA HDR(6)/4HX246/,HDR(2)/18/,HDR(1)/91678/
DATA HDR(5)/4HX149/,HDR(4)/19/,HDR(3)/91678/

```

```

C
REWIND OTP
WRITE(OTP) HDR
REWIND ITP
10 READ(ITP,1000) NWCO,NFLD
1000 FORMAT (A3,1X,I2)
IF(EOF(ITP)) 140,20
20 IF(NFLD.NE.1) GO TO 120
YTYP(1)=3
YTYP(2)=3
YTYP(3)=3
DO 30 I=4,74
YTYP(I)=0
30 YVAL(I)=0.0
YVAL(11)=FBLNK
YVAL(69)=FBLNK
40 READ(ITP,1010) XCOD,NFLD,ITAG,IBUF
1010 FORMAT (A3,1X,I2,1X,A1,2X,7A10)
IF(EOF(ITP)) 130,50
50 IF(XCOD.NE.NWCO) GO TO 110
IF(NFLD.LE.1) GO TO 120
IF(NFLD.GT.74) GO TO 120
IF(ITAG.NE.IBLNK) GO TO 60
YTYP(NFLD)=3
GO TO 70
60 IF(ITAG.NE.EST) GO TO 120
YTYP(NFLD)=2
70 CONTINUE
IF(NFLD.EQ.2) GO TO 80
IF(NFLD.EQ.3) GO TO 90
IF(NFLD.EQ.11) GO TO 100
IF(NFLD.EQ.69) GO TO 100
DECODE(16,1100,IBUF) YVAL(NFLD)
1100 FORMAT (G16.6)
GO TO 40
80 DECODE(40,1200,IBUF) YNAM
1200 FORMAT (5A8)
GO TO 40
90 DECODE(64,1300,IBUF) YPTH
1300 FORMAT (8A8)
GO TO 40
100 DECODE(8,1500,IRUF) YVAL(NFLD)
1500 FORMAT (A8)
GO TO 40
110 BACKSPACE ITP
WRITE(OTP) YTYP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)
GO TO 10
120 WRITE(6,1020)
1020 FORMAT (28H CONVERSION ERROR OCCURRED.)
STOP
130 WRITE(OTP) YTYP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)
140 ENDFILE OTP
REWIND OTP
REWIND ITP
WRITE(6,1030)
1030 FORMAT (24H SUCCESSFUL CONVERSION.)
STOP
END

```

7.1.5 Recognition Code/Model Cross-Reference

Three programs, Sections 7.1.5, 7.1.6 and 7.1.7, are all named program GAP, and contain many similarities; each has been adapted however to perform a different function.

In this section, the version of program GAP is used to read a HACS physical property data file and produce a cross-reference listing of chemical recognition codes which include a particular assessment model letter in the path codes on the file. The purpose of these programs was to obtain the cross-reference lists as quickly as possible. Since they will be run very infrequently, efficiency of internal operations was relatively unimportant, and large internal arrays are used to aggregate the required information.

As each chemical record is read, the hazard assessment model letters contained in the path codes on the file are individually stored in an array. A corresponding entry in a second array is made with the chemical recognition code. This process continues until all chemical data records have been read. The program currently allows a maximum of 3000 model references which is adequate for the 900 chemicals; however, this limit will need to be increased as new chemicals are added to the file.

When the input file has been completely processed, a second step is initiated. A single model letter is picked from the list A - Z, II, RR, SS, and the stored array of codes from the property file is searched for all occurrences of that letter. For each occurrence, the corresponding chemical recognition code is moved to another array for printing. The print array (SVCOD) is currently limited to a maximum of 900 entries, corresponding to the number of chemicals which is the maximum number of possible entries in the print array. This size will also need to be expanded as additional chemicals are added to the file.

After the stored code arrays have been entirely searched, the program prints the selected model letter, the number of references, and the list of chemical recognition codes (from the print array) that reference that model in their path codes. Since the alphabetic sequence of the original input file has been preserved, the chemical recognition codes are printed alphabetically within model letter codes.

The second step process is repeated for each different assessment model letter code until chemical recognition codes for all 29 models have been displayed.

81/09/18, 11.50.35.
PROGRAM RGPMD

PROGRAM GAP(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE9)

PROGRAM TABULATES CHEMICAL RECOGNITION CODES FOR
INDIVIDUAL RATE MODELS.

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.
35/318A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 03 JULY 1980

DIMENSION HDR(6),IPTH(16),PTLST(30),YNAM(5),YPTH(8),YTP(74)
DIMENSION YVAL(74)
INTEGER CLST(3000),MLST(3000),SVCOD(900)
INTEGER HDR,PTLST,YCOD,YTP
EQUIVALENCE (YCOD,YVAL(1))

DATA IBLNK/4H /,ITP/9/,LP/6/

ODATA (PTLST(I),I=1,30)/

1	4HA	4HB	4HC	4HD	4HE	4HF	4HG	4HH	,
2	4HI	4HJ	4HK	4HL	4HM	4HN	4HO	4HP	,
3	4HQ	4HR	4HS	4HT	4HU	4HV	4HW	4HX	,
4	4HY	4HZ	4HII	4HRR	4HSS	4H	/		

-----REWIND TAPE AND READ HEADER RECORD. TERMINATE IF GET END FILE.
REWIND ITP
READ(ITP) HDR
IF (EOF(ITP))5,30

-----INITIAL END OF FILE ERROR CONDITION
5 WRITE(LP,1010)
REWIND ITP
STOP

-----NORMAL RETURN
20 CONTINUE
WRITE(LP,1070) M
DO 24 I=1,29
MOD=PTLST(I)
N=0
DO 22 J=1,M
IF(MOD.NE.MLST(J)) GO TO 22
N=N+1
IF(N.GT.900) GO TO 70
SVCOD(N)=CLST(J)
22 CONTINUE
WRITE(LP,1080) MOD,N
WRITE(LP,1090) (SVCOD(K),K=1,N)
24 CONTINUE

REWIND ITP
STOP

-----PRINT REPORT TITLE AND DISPLAY FILE HEADER
30 CONTINUE
M=0
WRITE(LP,1050)
WRITE(LP,1020) HDR(5),HDR(4),HDR(3),HDR(6),HDR(2),HDR(1)

-----RETURN HERE TO READ NEXT PHYSICAL PROPERTY RECORD
40 READ(ITP) YTP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)

-----TEST FOR END OF FILE. STOP ON EOF.
IF(EOF(ITP)) 20,50

```

C
C-----PROCESS CHEMICAL RECORD
50 CONTINUE
C
C-----UNPACK PATH CODES AND STORE IN ARRAY IPTH
      DECODE(80,1040,YPTH) IPTH
      DO 60 I=1,16
      IF(IPTH(I).EQ.IBLNK) GO TO 60
      M=M+1
      IF(M.GT.3000) GO TO 70
      CLST(M)=YCD
      MLST(M)=IPTH(I)
60 CONTINUE
      GO TO 40
70 WRITE(LP,1060)
      REWIND ITP
      STOP
C
C
1010 FORMAT (/5X,46H*****ERROR - UNABLE TO READ HACS PROPERTY FILE)
10200FORMAT (/10X,21HFILE OPENED HAS ID = ,A4,20H, VERSION NUMBER = ,
1 15,10H, DATE = ,I6/13X,18HBACK-UP FILE ID = ,A4,20H, VERSION NU
2MBER = ,I5,10H, DATE = ,I6)
1030 FORMAT (2X,A3,3X,5A8,3X,8A8)
1040 FORMAT (8(2A4,2X))
1050 FORMAT (/5X,43HLIST OF CHEMICAL RECOGNITION CODES BY MODEL//)
1060 FORMAT (/5X,28H*****ERROR - BUFFER OVERFLOW)
1070 FORMAT (/5X,35HTOTAL NUMBER OF MODEL REFERENCES - ,I5)
10800FORMAT (/5X,6HMODEL ,A2,5H HAS ,I3,26H REFERENCES, BY CHEMICALS:
1 /)
1090 FORMAT (/14X,10A6)
      END
READY.

```

7.1.6 Recognition Code/Assessment Path Cross-Reference

The version of program GAP which follows performs an almost identical function to the version of program GAP in Section 7.1.5 except that the hazard assessment path codes contained on the input chemical properties file are evaluated in aggregate, not as single model letters. This identifies each different hazard assessment path code contained on the property file, and lists for each the chemical recognition codes of all chemicals giving the particular path code. Note that the hazard assessment scenarios are sub-sets of these path codes, and the information produced by this program was used to validate the rules of scenario formation developed for HACS/UIM.

The program reads each chemical data record and compares the path codes to a stored table of previously read path codes. If the path codes are found in the table, the chemical recognition code is stored in an array. A corresponding pointer is also set linking the recognition code to the table entry. If the path codes are not found in the table, they are appended to the end of the table as a new entry. The recognition code and linking pointer are stored as before. The program currently limits the size of this table to 100 different hazard assessment path codes; this should allow some expansion beyond the current 900 chemicals before an increase is required. However, the arrays of saved recognition codes and linking pointers are limited to 900 entries and expansion will be required as new chemicals are added to the file.

The second portion of the program simply loops through the stored table of different path codes. The table index is used to link to the stored array of chemical recognition codes to identify each chemical listing the indexed path code, and a printed display is produced.

Since the path code table is generated as the chemicals are processed, alphabetically by chemical recognition code, and since table entries are made using exact character matches, the resulting output report has the following characteristics:

- The path codes are listed in sequence according to the first appearance on the property file.
- Within each path code, chemicals are listed alphabetically by chemical recognition code.
- Logically equivalent path codes may appear more than once in the display containing the same model letter codes arranged in different sequence.

81/09/18. 11.51.33.
PROGRAM RGPPTH

PROGRAM GAP(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE9)

PROGRAM TO TABULATE UNIQUE PATH CODES ON FILE.

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.
35/318A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 03 JULY 1980

ODIMENSION HDR(6),IPTH(16),IREQ(74),LIST(14,74),LREF(30),PBUF(74),
1 PTLST(30),YNAM(5),YPTH(8),YTYP(74),YVAL(74)
DIMENSION NEST(74),NEXT(74),NGAP(74),NTOT(74)
INTEGER HDR,PBUF,PTLST,YCOD,YTYP
INTEGER SVCOD(900),SVPTH(900),PRCOD(900)
EQUIVALENCE (YCOD,YVAL(1))
INTEGER CNT(100),TOT
DIMENSION TAB(100,8)

DATA IBLNK/4H /,ITP/9/,LP/6/

ODATA (PTLST(I),I=1,30)/
1 4HA ,4HB ,4HC ,4HD ,4HE ,4HF ,4HG ,4HH ,
2 4HI ,4HJ ,4HK ,4HL ,4HM ,4HN ,4HO ,4HP ,
3 4HQ ,4HR ,4HS ,4HT ,4HU ,4HV ,4HW ,4HX ,
4 4HY ,4HZ ,4HTI ,4HRR ,4HSS ,4H /

-----REWIND TAPE AND READ HEADER RECORD. TERMINATE IF GET END FILE.
REWIND ITP
READ(ITP) HDR
IF (EOF(ITP))5,30

-----INITIAL END OF FILE ERROR CONDITION
5 WRITE(LP,1010)
STOP

-----NORMAL RETURN
20 CONTINUE
TOT=0
WRITE(LP,1090) N
1090 FORMAT (/5X,14HFILE CONTAINS ,I2,21H DIFFERENT PATH CODES/)
DO 21 I=1,N
TOT=TOT+CNT(I)
WRITE(LP,1070) CNT(I),(TAB(I,J),J=1,8)
1070 FORMAT (/5X,I3,25H OCCURRENCES OF PATH: ,8A8/)
L=0
DO 22 J=1,900
IF(SVPTH(J).NE.I) GO TO 22
L=L+1
PRCOD(L)=SVCOD(J)
22 CONTINUE
WRITE(LP,1100) (PRCOD(K),K=1,L)
1100 FORMAT (/10X,10A6)
21 CONTINUE
WRITE(LP,1080) TOT
1080 FORMAT (5X,5H-----/5X,I5)

REWIND ITP
STOP

-----PRINT REPORT TITLE AND DISPLAY FILE HEADER
30 CONTINUE
WRITE(LP,1050)
M=0
N=0

```

DO 25 I=1,100
25 CNT(I)=0
WRITE(LP,1020) HDR(5),HDR(4),HDR(3),HDR(6),HDR(2),HDR(1)
C-----RETURN HERE TO READ NEXT PHYSICAL PROPERTY RECORD
40 READ(ITP) YTP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)
C-----TEST FOR END OF FILE. STOP ON EOF.
IF(EOF(ITP)) 20,50
C-----PROCESS CHEMICAL RECORD
50 CONTINUE
C-----COUNT IF PATH ALREADY IN TABLE, OTHERWISE ADD TO TABLE.
IF(N.EQ.0) GO TO 80
DO 70 I=1,N
DO 60 J=1,8
IF(YPTH(J).NE.TAB(I,J)) GO TO 70
60 CONTINUE
M=M+1
SVCOD(M)=YCOD
SVPTH(M)=I
CNT(I)=CNT(I)+1
GO TO 40
70 CONTINUE
80 N=N+1
IF(N.GT.100) GO TO 100
DO 90 J=1,8
90 TAB(N,J)=YPTH(J)
CNT(N)=CNT(N)+1
M=M+1
SVCOD(M)=YCOD
SVPTH(M)=N
GO TO 40
100 WRITE(LP,1060)
STOP
C
1010 FORMAT (/5X,46H****ERROR - UNABLE TO READ HACS PROPERTY FILE)
10200FORMAT (/10X,21HFILE OPENED HAS ID = ,A4,20H, VERSION NUMBER : ,
1 IS,10H, DATE = ,I6/13X,18HBACK-UP FILE ID = ,A4,20H, VERSION NU
2MBER = ,I5,10H, DATE = ,I6)
1030 FORMAT (2X,A3,3X,5A8,3X,8A8)
1040 FORMAT (8(2A4,2X))
1050 FORMAT (/5X,26HTABLE OF UNIQUE PATH CODES//)
1060 FORMAT (/5X,14HTABLE OVERFLOW)
END
READY.

```

7.1.7 Data Gap Identification

The version of program GAP which follows is the master chemical property file data gap identification program which correlates data values missing on the property file with actual HACS model input requirements.

Matrix elements are set in the program to define the correspondence between HACS model inputs and individual property data items. Note that the program cannot distinguish among different conditions which may occur within a rate model, that is, property input required under some conditions but not others.

The following program listing gives detailed documentation. Note that the program can also be modified (refer to comments in the listing) to perform selective screening functions and these steps were used to develop the overall data gap analyses.

81/09/18. 11.52.38.
PROGRAM RGP GAP

PROGRAM GAP(INPUT,OUTPUT,TAPE4=OUTPUT,TAPE9)

PROGRAM GAP READS AS INPUT A HACS PHYSICAL PROPERTY DATA FILE (TAPE9) AND PRODUCES AS OUTPUT A PRINTED REPORT IDENTIFYING ALL DATA GAPS BY PROPERTY ITEM FIELD NUMBER. A DATA GAP IS DEFINED AS A PROPERTY ITEM WHICH MAY BE REQUIRED BY AT LEAST ONE OF THE MODELS IN THE PATH CODE ON THE FILE AND WHICH HAS A CURRENT STATUS TAG OF MISSING. CORRESPONDENCE BETWEEN THE CURRENT HACS MODELS AND PROPERTY DATA ITEMS IS GIVEN BY THE ELEMENTS IN THE DATA ARRAY LIST. ANY MODEL CHANGES OR REVISIONS RELATED TO THE USE OF PROPERTY DATA MAY ALSO REQUIRE UPDATES TO THIS ARRAY.

THE ARRAY LREF CONTROLS THE INDEXING FROM AN ALPHABETIC MODEL CODE, OBTAINED FROM THE INPUT PROPERTY FILE IN THE ARRAY IPTH, TO THE APPROPRIATE ENTRY IN THE LIST OF REQUIRED ITEMS, LIST. THE PROGRAM CAN BE ADAPTED FOR SELECTIVE DATA GAP SCREENING BY MODIFYING DATA VALUES IN THE ARRAY LREF. A VALUE IN LREF OF 0 INDICATES THE CORRESPONDING MODEL EITHER DOES NOT REQUIRE PROPERTY DATA, OR HAS BEEN OMITTED FROM THE SCREEN.

THE PROGRAM MAY ALSO BE ADAPTED FOR SELECTIVE SCREENING BY SPECIFIC PROPERTY ITEMS BY EITHER MODIFYING THE ELEMENTS OF THE ARRAY LIST WHICH ARE SET TO 1, OR BY ALLOWING ONLY SELECTED VALUES OF IREQ TO BE SET TO 1 IN THE DO LOOP ON J TO STATEMENT 90. FOR EXAMPLE, THIS COULD BE USED TO ISOLATE ALL ACTUAL DATA GAPS FOR A SINGLE TEMPERATURE FUNCTION.

AFTER PRINTING IDENTIFICATION OF ALL DATA GAPS FOR EACH CHEMICAL ON THE FILE, THE PROGRAM PRINTS SUMMARY COUNTS OF THE NUMBER OF PROPERTY ITEMS ACTUALLY USED BY THE MODELS, BROKEN DOWN BY STATUS CODE. NOTE THAT THESE SUMMARY COUNTS ARE NOT THE SAME AS GIVEN BY THE PROPERTY RETRIEVAL PROGRAM (WHICH REPORTS ON ALL ITEMS STORED, WHETHER OR NOT USED).

HDR = HEADER, FIRST RECORD ON PROPERTY FILE, IDENTIFIES
CURRENT AND PREVIOUS FILE VERSIONS
I = LOOP INDEX
IBLNK = DATA WORD SET TO BLANKS USED TO SKIP EMPTY MODEL
CODES IN ASSESSMENT PATH DATA
IPTH = PATH CODES FOR SINGLE CHEMICAL, UP TO 16 MODEL CODES
INTERPRETED FROM INPUT ARRAY YPTH TO GIVE SINGLE
ALPHABETIC MODEL CODE IN EACH WORD
IREQ = CONTROLLING ARRAY FOR DATA GAP SCREEN FOR SINGLE
CHEMICAL, ELEMENT I IS INITIALIZED TO ZERO, THEN
SET TO 1 IF ANY MODEL IN PATH CODE LIST FOR THE
CHEMICAL USES PROPERTY ITEM I AS INPUT.
ITP = FORTRAN UNIT REFERENCE NUMBER FOR INPUT PROPERTY
TAPE FILE
J = LOOP INDEX
K = INDEX POINTER TO LINE OF ARRAY LIST FOR SINGLE
MODEL CODE IN PATH LIST, OR ZERO IF MODEL DOES
NOT REQUIRE PROPERTY DATA
LGAP = SEQUENCE COUNT OF DATA GAPS FOUND FOR SINGLE CHEMICAL
LIST = MASTER REFERENCE LIST ESTABLISHING EACH UNIQUE SET
OF PROPERTY DATA ITEMS, CURRENTLY CONTAINS 14 SUB-
LISTS OF 74 ELEMENTS EACH. SOME SUB-LISTS ARE USED
MORE THAN ONCE (FOR DIFFERENT MODELS). ELEMENT I
ON SUB-LIST IS SET TO 1 IF MODEL CORRESPONDING TO
SUB-LIST USES PROPERTY ITEM I, OR IS ZERO OTHERWISE.
LP = FORTRAN UNIT REFERENCE NUMBER FOR PRINTER
LREF = ARRAY CONTAINING SUB-LIST INDEX NUMBER (TO ARRAY LIST)
FOR EACH ALPHABETIC MODEL CODE, IN POSITION
CORRESPONDING TO LOCATION OF MODEL CODE IN PTLS
NEST = NUMBER OF REQUIRED PROPERTY ITEMS ON DATA FILE FOR
WHICH THE CURRENT VALUE IS AN ESTIMATE
NEXT = NUMBER OF REQUIRED PROPERTY ITEMS ON DATA FILE FOR

WHICH THE CURRENT VALUE IS EXACT
 NGAP = NUMBER OF REQUIRED PROPERTY ITEMS ON DATA FILE FOR
 WHICH THE CURRENT VALUE IS MISSING
 NTOT = TOTAL NUMBER OF ITEMS ON PROPERTY FILE ACTUALLY
 USED BY HACS MODEL.
 PBUF = PRINT BUFFER, ELEMENTS 1 TO LGAP CONTAIN PROPERTY
 ITEM INDEX NUMBER OF EACH DATA GAP FOR SINGLE
 CHEMICAL
 PTLST = LIST OF ALL ALPHABETIC MODEL LETTER CODES WHICH CAN
 APPEAR IN PATH CODE FOR CHEMICAL
 YCOD = CHEMICAL RECOGNITION CODE READ FROM PROPERTY FILE
 YNAM = CHEMICAL NAME
 YPTH = ARRAY OF MODEL CODES READ ON INPUT FOR PATH CODE FOR
 PARTICULAR CHEMICAL. CONVERSION FROM CDC 3300
 PRODUCES STORAGE OF MORE THAN ONE CODE PER WORD OF
 ARRAY. CONVERSION IN THIS PROGRAM PRODUCES USFABLE
 LIST OF CODES IN ARRAY IPTH.
 YTPP = ARRAY OF STATUS CODES FOR EACH ITEM (1 TO 74) STORED
 ON PROPERTY FILE FOR SINGLE CHEMICAL (0=MISSING,
 2=ESTIMATE, 3=EXACT)
 YVAL = ARRAY USED FOR STORAGE OF NUMERIC VALUES OF PROPERTY
 ITEMS, REQUIRED HERE ONLY TO READ ENTIRE CHEMICAL
 DATA RECORD

COMMON VARIABLES USED - NONE

SUBROUTINES REQUIRED - NONE

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.
 35/318A ACORN PARK,
 CAMBRIDGE, MASS., 02140
 TEL. 617-864-5770 EXT. 2813

DATE - 03 JULY 1980

ODIMENSION HDR(6),IPTH(16),IREQ(74),LIST(14,74),LREF(30),PBUF(74),
 1 PTLST(30),YNAM(5),YPTH(8),YTPP(74),YVAL(74)
 DIMENSION NEST(74),NEXT(74),NGAP(74),NTOT(74)
 INTEGER HDR,PBUF,PTLST,YCOD,YTPP
 EQUIVALENCE (YCOD,YVAL(1))

DATA IBLNK/4H /,ITP/9/,LP/6/

ODATA (LIST(1,I),I=1,74)/3*0,2*1,6*0,5*1,14*0,4*1,8*0,3*1,2*0,6*1,
 1 0,1,19*0/
 DATA (LIST(2,I),I=1,74)/3*0,1,38*0,3*1,24*0,4*1,0/
 DATA (LIST(3,I),I=1,74)/3*0,1,56*0,1,2*0,1,10*0/
 DATA (LIST(4,I),I=1,74)/4*0,1,6*0,5*1,2*0,4*1,32*0,1,19*0/
 DATA (LIST(5,I),I=1,74)/3*0,2*1,6*0,5*1,26*0,3*1,17*0,1,9*0,1,0/
 DATA (LIST(6,I),I=1,74)/4*0,1,6*0,5*1,18*0,1,7*0,3*1,9*0,1,19*0/
 DATA (LIST(7,I),I=1,74)/3*0,2*1,0,1,4*0,5*1,22*0,1,0,5*1,29*0/
 DATA (LIST(8,I),I=1,74)/3*0,2*1,6*0,5*1,44*0,1,2*0,1,10*0/
 DATA (LIST(9,I),I=1,74)/3*0,2*1,0,1,4*0,5*1,58*0/
 ODATA (LIST(10,I),I=1,74)/3*0,2*1,0,1,4*0,5*1,26*0,3*1,15*0,1,2*0,
 1 1,10*0/
 ODATA (LIST(11,I),I=1,74)/3*0,2*1,0,1,4*0,5*1,2*0,4*1,12*0,1,0,1,0,
 1 1,0,2*1,32*0/
 ODATA (LIST(12,I),I=1,74)/3*0,2*1,6*0,5*1,2*0,4*1,8*0,4*1,2*0,1,
 1 5*0,3*1,9*0,1,5*0,1,13*0/
 DATA (LIST(13,I),I=1,74)/3*0,1,7*0,5*1,44*0,1,2*0,1,10*0/
 ODATA (LIST(14,I),I=1,74)/3*0,2*1,0,1,4*0,5*1,20*0,1,0,1,0,2*1,
 1 32*0/

ODATA (LREF(I),I=1,30)/1,2,3,4,5,4,3,5,6,3,7,5,0,8,0,9,5,10,3,
 1 11,5,12,13,14,6*0/

ODATA (PTLST(I),I=1,30)/
 1 4HA ,4HB ,4HC ,4HD ,4HE ,4HF ,4HG ,4HH ,
 2 4HI ,4HJ ,4HK ,4HL ,4HM ,4HN ,4HO ,4HP ,
 3 4HQ ,4HR ,4HS ,4HT ,4HU ,4HV ,4HW ,4HX ,
 4 4HY ,4HZ ,4HII ,4HRR ,4HSS ,4H /

```

C
C
C-----INITIALIZE SUMMARY COUNTS
      DO 1 I=1,74
        NEST(I)=0
        NEXT(I)=0
        NGAP(I)=0
      1 NTOT(I)=0
C
C-----REWIND TAPE AND READ HEADER RECORD.  TERMINATE IF GET END FILE.
      REWIND ITP
      READ(ITP) HDR
      IF (EOF(ITP))5,30
C
C-----INITIAL END OF FILE ERROR CONDITION
      5 WRITE(LP,1010)
C-----NORMAL RETURN
      20 CONTINUE
C
C-----PRINT SUMMARY COUNTS OF REQUIRED DATA ITEMS
      WRITE(LP,1090)
      DO 25 I=4,74
        WRITE(LP,1100) I,NGAP(I),NEST(I),NEXT(I),NTOT(I)
        NGAP(I)=NGAP(I)+NGAP(I)
        NEST(I)=NEST(I)+NEST(I)
        NEXT(I)=NEXT(I)+NEXT(I)
      25 NTOT(I)=NTOT(I)+NTOT(I)
      WRITE(LP,1110) NGAP(1),NEST(1),NEXT(1),NTOT(1)
      REWIND ITP
      STOP
C
C-----PRINT REPORT TITLE AND DISPLAY FILE HEADER
      30 CONTINUE
      WRITE(LP,1120)
      WRITE(LP,1020) HDR(5),HDR(4),HDR(3),HDR(6),HDR(2),HDR(1)
C
C-----RETURN HERE TO READ NEXT PHYSICAL PROPERTY RECORD
      40 READ(ITP) YTP,YVAL(1),YNAM,YPTH,(YVAL(I),I=4,74)
C
C-----TEST FOR END OF FILE.  STOP ON EOF.
      IF(EOF(ITP)) 20,50
C
C-----PROCESS CHEMICAL RECORD
      50 CONTINUE
C
C-----UNPACK PATH CODES AND STORE IN ARRAY IPTH
      DECODE(80,1040,YPTH) IPTH
C
C-----INITIALIZE ARRAY IREQ
      DO 60 I=1,74
        60 IREQ(I)=0
C
C-----LOOP THROUGH EACH MODEL CODE IN PATH READ FOR SINGLE CHEMICAL.
      SKIP ALL BLANK MODEL CODES IN PATH.
      DO 100 I=1,16
        IF(IPTH(I).EQ.IBLNK) GO TO 100
C
C-----FOR EACH NON-BLANK MODEL CODE READ, USE LIST OF VALID CODES
      TO TRANSLATE FROM LETTER CODE TO NUMERIC SUB-LIST INDEX K.
      K=0
      DO 70 J=1,29
        IF(IPTH(I).NE.PTLST(J)) GO TO 70
C-----FOUND MATCH, CROSS-REFERENCE FROM MODEL LETTER CODE TO
      SUB-LIST OF REQUIRED PROPERTY DATA.
      K=LREF(J)
      GO TO 80
      70 CONTINUE
C
C-----NOTE - K=0 HERE IF MODEL DOES NOT USE PROPERTY DATA OR IF
      MODEL CODE IS NOT DEFINED ON LIST.  SKIP IF NO DATA IS REQUIRED
      FOR THIS MODEL.
      80 IF(K.EQ.0) GO TO 100

```

```

C
C-----SET EACH ELEMENT OF IREQ ARRAY TO 1 FOR EACH REQUIRED DATA
C      ITEM FOR THIS MODEL CODE.
      DO 90 J=1,74
      IF(LIST(K,J).GT.0) IREQ(J)=1
      90 CONTINUE
C
C-----CONTINUE LOOP FOR EACH MODEL CODE IN PATH
      100 CONTINUE
C
C-----FOR EACH REQUIRED DATA ITEM, CHECK STATUS CODE.  STORE INDEX
C      OF MISSING ITEMS IN ARRAY PBUF FOR PRINTING.
      LGAP=0
      DO 110 I=1,74
      IF(IREQ(I).EQ.0) GO TO 110
C-----COUNT TOTAL NUMBER OF DATA ITEMS ACTUALLY REQUIRED.
      NTOT(I)=NTOT(I)+1
C-----BRANCH ON STATUS CODE OF ITEM.  INCREMENT COUNTS OF EXACT AND
C      ESTIMATED ITEMS THEN CONTINUE IF VALUE IS NOT MISSING.
      IF(YTYP(I).EQ.3) NEXT(I)=NEXT(I)+1
      IF(YTYP(I).EQ.2) NEST(I)=NEST(I)+1
      IF(YTYP(I).GT.0) GO TO 110
C-----ITEM IS DATA GAP, STORE ITEM INDEX FOR PRINT DISPLAY
      NGAP(I)=NGAP(I)+1
      LGAP=LGAP+1
      PBUF(LGAP)=I
      110 CONTINUE
C
C-----SKIP IF NO DATA GAPS, OTHERWISE DISPLAY CHEMICAL ID FIRST.
C      PBUF DISPLAY IS FRAGMENTED TO AVOID UNNECESSARY BLANK LINES
C      IN OUTPUT OF VARIABLE LENGTH DATA WHICH MAY POSSIBLY EXCEED
C      LENGTH OF ONE OR TWO PRINT LINES.  AFTER DISPLAY, CYCLE
C      BACK FOR NEXT CHEMICAL OR EOF.
      IF(LGAP.EQ.0) GO TO 40
      WRITE(LP,1030) YCOD,YNAM,YPTH
      120 IF(LGAP.GT.30) GO TO 130
      WRITE(LP,1070) LGAP,(PBUF(I),I=1,LGAP)
      GO TO 40
      130 WRITE(LP,1070) LGAP,(PBUF(I),I=1,30)
      IF(LGAP.GT.60) GO TO 140
      WRITE(LP,1080) (PBUF(I),I=31,LGAP)
      GO TO 40
      140 WRITE(LP,1080) (PBUF(I),I=31,60)
      WRITE(LP,1080) (PBUF(I),I=61,LGAP)
      GO TO 40
C
1010 FORMAT (/5X,46H****ERROR - UNABLE TO READ HACS PROPERTY FILE)
10200FORMAT (/10X,21HFILE OPENED HAS ID = ,A4,20H, VERSION NUMBER = ,
1 15,10H, DATE = ,16/13X,19HBACK-UP FILE ID = ,A4,20H, VERSION NU
2MBER = ,15,10H, DATE = ,16)
1030 FORMAT (2X,A3,3X,5A8,3X,8A8)
1040 FORMAT (8(2A4,2X))
1070 FORMAT (5X,I2,20H DATA GAPS, ITEMS = ,30I3)
1080 FORMAT (27X,30I3)
10900FORMAT (////9X,39HHACS PHYSICAL PROPERTY DATA GAP SUMMARY/9X,
1 39(1H-)//5X,44HFIELD MISSING ESTIMATED EXACT TOTAL/5X,
2 46HNUMBER VALUES VALUES VALUES REQUIRED/5X,6(1H-),3X,
3 7(1H-),2X,9(1H-),2X,6(1H-),3X,8(1H-))
1100 FORMAT (7X,I2,4(5X,I5))
1110 FORMAT (5X,6(1H-),4(3X,5(1H-),2X)/5X,5HTOTAL,4(3X,I6,1X)////)
1120 FORMAT (5X,38HHACS PHYSICAL PROPERTY DATA GAP REPORT////)
      END
READY.

```

7.2 Message File Creation

Section 7.2 contains listings of three programs used to create the field text, scenario descriptions and model explanations. The original version of the program was created to process messages entered for HACS data field explanations; the following two versions were then created as special cases or simplifications. Corresponding to the file load programs in this section are three message display programs given in Section 7.3.

The programs^c used to create the field text message file were prepared to provide for coded messages, and control for either interactive or batch processing. This allows for both the initial file creation step, as well as any subsequent editing which may be desired.

The file of scenario text data was prepared as uncoded messages using a similar version of the program. However, the need for changing these messages is probably limited.

The file of model text data was prepared using a simplified version of the program to process model text input in batch mode only.

Note that maximum message lengths, number of messages, format control characters, and related message attributes are different among these three programs and resulting files.

7.2.1 Field Text (File RGPTXT)

PROGRAM TXTLOD(INPUT,OUTPUT,TAPES=INPUT,TAPE6=OUTPUT,
1 TAPE10,TAPE11,TAPE12)

PROGRAM LOADS A MASS STORAGE FILE WITH CODED MESSAGE TEXT DATA, INDEXED BY MESSAGE NUMBER. TWO MODES OF OPERATION ARE PROVIDED: INTERACTIVE AND BATCH. THE FILE IS INITIALLY BUILT IN THE INTERACTIVE MODE IN WHICH A SERIES OF TEXT MESSAGES ARE ADDED TO THE PREVIOUS CONTENTS OF THE MS FILE. MESSAGES ARE ADDED IN SEQUENCE BY MESSAGE NUMBER IMMEDIATELY FOLLOWING THE LAST MESSAGE PREVIOUSLY ENTERED. INTERACTIVE RUNS MAY BE TERMINATED AND RE-STARTED AT A LATER TIME UNTIL THE FULL 256 MESSAGE FILE IS CREATED. IN BATCH MODE, A TEXT FILE, PRODUCED BY OUTPUT FROM A SEPARATE PROGRAM FOR ON-LINE TEXT EDITING, CONTAINING THE FULL SET OF TEXT MESSAGES IS PROCESSED FROM START TO FINISH. IN GENERAL, ERRORS ENCOUNTERED IN THE BATCH MODE ARE IMMEDIATELY FATAL, WHILE THE PROGRAM PERMITS RECOVERY FROM MOST INTERACTIVE ERRORS.

THE HACS DEFAULT FILE IS READ BY THIS PROGRAM TO OBTAIN FIELD NUMBERS, NAMES AND VARIABLE TYPES FOR USE IN DISPLAYS AND TO CORRELATE THE SEQUENTIAL MESSAGE NUMBERS (FROM 1 TO 256) WITH FIELD POSITIONS IN THE DEFAULT FILE.

THE MASS STORAGE FILE PRODUCED BY THIS PROGRAM CONTAINS VARIABLE LENGTH RECORDS, INDEXED NUMERICALLY BY A MESSAGE NUMBER WHICH RANGES FROM 1 TO 256. A SINGLE RECORD IS CREATED IN THE FILE FOR EACH MESSAGE NUMBER. RECORDS IN THE FILE ARE CODED AS FOLLOWS:

- UNCODED - UNCODED RECORDS CONTAIN TEXT (SEE BELOW)
- TYPE 1 - TYPE 1 RECORDS CONTAIN 3 WORDS OF 10 CHARACTERS TO CONSTRUCT THE MINIMUM RECORD SIZE FOR A CDC MS FILE. THE FIRST WORD CONTAINS 1H1, FOLLOWED BY BLANKS; THE SECOND AND THIRD WORDS ARE ALSO BLANK. TYPE 1 MESSAGES ARE STANDARDIZED IN HACS AS REFERENCES TO THE USER MANUAL.
- TYPE 2 - THESE ARE SIMILAR TO TYPE 1 RECORDS, BUT CONTAIN 1H2 FOLLOWED BY BLANKS. TYPE 2 MESSAGES ARE STANDARDIZED IN HACS AS REFERENCES TO CHEMICAL PROPERTY DATA.
- TYPE 3 - THESE ARE SPECIAL TYPE 2 RECORDS FOR WHICH ADDITIONAL TEXT IS ALSO GIVEN. THE FIRST WORD OF THE RECORD CONTAINS 1H3 FOLLOWED BY BLANKS. THE REMAINDER OF THE RECORD CONTAINS VARIABLE LENGTH TEXT.

TEXT CONTAINED IN UNCODED OR TYPE 3 RECORDS IS VARIABLE LENGTH (RECORD LENGTH GE 3), AND PACKED AS 10 CHARACTERS PER WORD. RECORDS CONTAIN ONE OR MORE LINES OF MESSAGE TEXT ORIGINALLY ENTERED VIA THIS PROGRAM IN INTERACTIVE MODE. EMBEDDED BLANKS BETWEEN LINES ARE AUTOMATICALLY REMOVED, AND FORTRAN FORMAT CODE FOR "/SX," IS AUTOMATICALLY INSERTED BETWEEN CHARACTER STRINGS REPRESENTING DIFFERENT LINES. WHEN USED BY HACS, THE MESSAGE DATA IS READ INTO AN ARRAY AND FORMAT CODE APPENDED TO THE BEGINNING AND END OF THE ARRAY. THE EXPANDED ARRAY IS THEN USED AS AN EXECUTION TIME FORMAT TO PRODUCE THE DESIRED MESSAGE. THIS PROCEDURE IS ILLUSTRATED IN THE AUDIT PORTION OF THE PROGRAM BELOW.

SINCE THE MESSAGE TEXT IS STORED AND USED AS PACKED CHARACTER DATA, THIS PROGRAM IS SIGNIFICANTLY MACHINE DEPENDENT. OUTPUT PRODUCED BY THIS PROGRAM IS WRITTEN AS 10 CHARACTER WORDS. IN ADDITION TO THE USE OF OPEN, WRITE AND CLOSE MASS STORAGE UTILITY SUBROUTINES, THE CDC ENCODE FUNCTION IS USED TO PACK MESSAGE DATA READ IN A1 FORMAT TO A10 FORMAT FOR OUTPUT TO THE MS FILE.

BUFF = A10 STORAGE FOR USER RESPONSE TO VERIFICATION QUERY. ANY RESPONSE OTHER THAN CARRIAGE RETURN REJECTS MESSAGE INPUT.
 DEFLT = EXTERNAL HACS DEFAULT FILE, READ TO OBTAIN HACS FIELD NUMBER, FIELD NAME AND FIELD TYPE (INTEGER OR REAL) CORRESPONDING TO SEQUENTIAL FIELD MESSAGE NUMBERS.
 FORM = SIX-WORD ARRAY CONTAINING FORMAT CODE TO BE INSERTED BETWEEN EACH LINE OF MULTIPLE LINE MESSAGES.
 FPCK = ARRAY EQUIVALENCED TO PACKED MESSAGE TEXT, AND CONTAINING STANDARD FORMAT CODE IN FIRST WORD
 FVAL = ARRAY OF DEFAULT VALUES, MINIMUM AND MAXIMUM FOR REAL HACS DATA FIELDS. READ AS PART OF DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 I = GENERAL LOOP INDEX
 IO = FIRST CHARACTER OF ARRAY TEXT FOR START OF PACK INTO ARRAY PACK, SET TO 1 FOR UNCODED MESSAGES OR TO 2 FOR TYPE 3 MESSAGES.
 IBLNK = DATA WORD SET TO ALL BLANKS FOR USE IN INITIALIZING CHARACTER VARIABLES
 IBTCH = DEVICE NUMBER FOR BATCH INPUT FILE WHEN RUNNING IN BATCH MODE
 IN = INPUT DEVICE NUMBER SET TO EITHER TTY OR IBTCH VIA USER CONTROL AT START OF RUN
 ISTRT = INDEX NUMBER OF LAST MESSAGE PREVIOUSLY LOADED ONTO MASS STORAGE FILE. EACH RUN OF PROGRAM IN THE INTERACTIVE MODE ADDS MESSAGES SEQUENTIALLY BY 1 FROM ISTRT+1 UP TO A MAXIMUM OF 256. IN BATCH MODE, A BULK RE-CREATION OF THE MS FILE BUILDS ALL MESSAGES FROM 1 TO 256.
 ISW = CONTROL FLAG SET TO 0 FOR INTERACTIVE INPUT, OR TO 1 FOR BATCH INPUT.
 IVAL = ARRAY OF DEFAULT VALUES, MINIMUM AND MAXIMUM FOR INTEGER HACS DATA FIELDS. READ AS PART OF DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 IVAR = CODED FIELD TYPE INDICATOR PACKED IN ARRAY LIST FOR EACH HACS DATA FIELD (0=INTEGER FIELD, 1=REAL FIELD)
 J = GENERAL LOOP INDEX
 = INDEX TO LAST NON-BLANK CHARACTER READ ON SINGLE MESSAGE LINE
 = WORD INDEX TO ARRAY PACK, AND LENGTH OF FINAL PACKED MESSAGE TEXT (IN WORDS)
 JP = DUMMY INDEX, SET TO J+1, USED TO STORE FORMAT TERMINATOR IN PACKED MESSAGE TEXT FOR DISPLAY BUT NOT OUTPUT TO MS FILE.
 LEN = COUNT OF CHARACTERS IN TEXT TO BE MOVED TO ARRAY PACK IN A10 FORMAT
 LINE = BUFFER USED FOR STORAGE OF SINGLE INPUT LINE OF MESSAGE TEXT IN A1 FORMAT. LENGTH SET TO 130 CHARACTERS FOR USE WITH WIDE-BODY TERMINALS, HOWEVER ACTUAL MESSAGE LENGTH IS LIMITED TO 70 CHARACTERS PER LINE.
 LIST = ARRAY OF CODES IN HACS DEFAULT FILE DEFINING STRUCTURE OF DATA FIELD ITEM I AS LIST(I,J) WHERE J=1,6. LIST(I,1) GIVES FIELD NUMBER FOR FIELD I AND LIST(I,J), J=3,5, GIVES FIELD NAME. ELEMENT 2 CONTAINS CODED SOURCE CODE, VARIABLE TYPE AND QUANTITY TYPE. ELEMENT 6 IS INDEX TO STORAGE OF NUMERIC VALUES IN FVAL OR IVAL. REFER TO HACS PROGRAM DOCUMENTATION FOR COMPLETE DETAILS.
 LP = UNIT DEVICE FOR PROGRAM OUTPUT IN EITHER BATCH OR INTERACTIVE MODES
 M = COUNT OF TOTAL NUMBER OF MESSAGE CHARACTERS ACCUMULATED IN ARRAY TEXT
 MNF = MAXIMUM NUMBER OF REAL FIELD ITEMS ALLOWED IN HACS DEFAULT FILE. READ FROM DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 MNI = MAXIMUM NUMBER OF INTEGER FIELD ITEMS ALLOWED IN HACS DEFAULT FILE. READ FROM DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.


```

C-----READ EXTERNAL DEFAULT FILE CONTAINING FIELD NUMBERS AND
C NAMES INDEXED SEQUENTIALLY.
  REWIND UDFLT
  READ(UDFLT) DEFLT
C
C-----PROGRAM ASSUMES DEFAULT FILE CONTAINS EXACTLY 256 ENTRIES.
  NFLD=NFLD+1
  IF(NFLD,EQ,256) GO TO 50
  WRITE(LP,1020)
  STOP
50 CONTINUE
C
C-----OPEN MASS STORAGE MESSAGE FILE, INDEXED BY MESSAGE NUMBER
  CALL OPENMS(UTXT,NDEX,257,0)
C
C-----READ AND VALIDATE NUMBER OF LAST MESSAGE ENTERED TO INITIALIZE
C START OF SEQUENCE.
60 IF(ISW,EQ,0) WRITE(TTY,1030)
  IF(EOF(IN)) 70,70
70 READ(IN,1040) ISTRT
  IF(ISTRT,LT,0) GO TO 80
  IF(ISTRT,LE,256) GO TO 90
80 WRITE(LP,1050)
  IF(ISW,EQ,0) GO TO 60
  STOP
90 CONTINUE
C
C-----START OF EACH NEW CYCLE FOR NEXT MESSAGE. CLOSE FILE AND STOP
C AFTER 256 MESSAGES ARE STORED.
100 ISTRT=ISTRT+1
  IF(ISTRT,LE,256) GO TO 120
110 CALL CLOSMS(UTXT)
  STOP
120 CONTINUE
C
C-----READ SINGLE LINE FROM BATCH INPUT FILE (SKIP FOR INTERACTIVE
C INPUT). LINE IN FILE GIVES MESSAGE NUMBER, HACS FIELD NUMBER
C AND HACS FIELD NAME FOR REFERENCE USE. ONLY THE MESSAGE
C NUMBER IS READ AND MUST MATCH THE SEQUENCE COUNT ISTRT.
  IF(ISW,EQ,0) GO TO 130
  READ(IN,1040) NMSG
  IF(NMSG,EQ,ISTRT) GO TO 130
  WRITE(LP,1060) NMSG
  STOP
C
C-----WRITE PROMPT DISPLAY FOR EITHER INTERACTIVE OR BATCH INPUT
C GIVING THE MESSAGE NUMBER, FIELD MODE, NUMBER AND NAME.
130 IVAR=LIST(ISTRT,2)/1000
  IVAR=IVAR+1
  OWRITE(LP,1070) ISTRT,(MOD(IVAR,1),I=1,2),LIST(ISTRT,1),
  1 (LIST(ISTRT,J),J=3,5)
C
C-----INITIALIZE FOR ACCUMULATION OF MULTIPLE LINE MESSAGES
  M=0
  DO 140 I=1,684
140 TEXT(I)=IBLNK
  DO 150 I=1,70
150 PACK(I)=IBLNK
  NLIN=0
C
C-----RETURN HERE TO READ EACH NEW LINE OF MULTIPLE LINE MESSAGE,
C OR BLANK LINE TERMINATING MESSAGE. INITIALIZE.
160 NLIN=NLIN+1
170 DO 180 I=1,130
180 LINE(I)=IBLNK
C
C-----READ INPUT LINE OF UP TO 130 CHARACTERS IN A1 FORMAT. THE
C FORMAT FOR INTERACTIVE INPUT SUPPRESSSES LEADING SPACES. FOR
C BATCH INPUT, THE TEXT IS PRECEDED BY FIVE SPACES SINCE THE
C BATCH FILE CREATED FOR EDITING IS PREPARED BY EXECUTING THE
C PREVIOUS MESSAGES AS FORMAT STATEMENTS.
  IF(EOF(IN)) 190,190

```

```

190 CONTINUE
   IF(ISW.EQ.0) READ(IN,1080) LINE
   IF(ISW.EQ.1) READ(IN,1090) LINE
C-----LOCATE THE LAST NON-BLANK CHARACTER READ ON THE INPUT LINE.
   DO 200 I=1,130
     J=131-I
     IF(LINE(J).NE.IBLNK) GO TO 210
200 CONTINUE
     J=0
210 CONTINUE
C-----J GIVES THE INDEX TO THE LAST NON-BLANK CHARACTER READ ON
C      INPUT.  VALID TEXT LINES MAY BE BLANK IF NOT THE FIRST LINE,
C      AND MAY NOT CONTAIN MORE THAN 70 CHARACTERS OF TEXT.
     IF(J.LE.70) GO TO 220
C-----SINGLE LINE OVERFLOW.
     WRITE(LP,1100) NLIN
     IF(ISW.EQ.1) STOP
     GO TO 170
C-----TEST FOR LAST LINE (BLANK) TERMINATING MESSAGE.
220 IF(J.GT.0) GO TO 230
C-----IF BLANK LINE IS PRECEDED BY TEXT, HAVE VALID END OF MESSAGE.
C      BRANCH TO PROCESS.  OTHERWISE, HAVE ERROR.
     IF(M.GT.0) GO TO 270
     WRITE(LP,1110)
     IF(ISW.EQ.1) STOP
     GO TO 170
C-----GET HERE WITH MESSAGE TEXT ENTERED FOR SINGLE LINE.  TEXT
C      MESSAGES ARE ALLOWED FOR UP TO 9 NON-BLANK LINES OF TEXT.  AN
C      ERROR OCCURS IF MORE THAN 9 LINES ARE ENTERED.
230 IF(NLIN.LE.9) GO TO 240
     WRITE(LP,1120)
     IF(ISW.EQ.1) STOP
     GO TO 130
C-----APPEND NEW LINE OF TEXT TO ACCUMULATED MESSAGE TEXT, INSERT
C      OUTPUT FORMAT CONTROL AT END OF EACH NEW LINE, THEN RETURN TO
C      READ NEXT TEXT LINE OR BLANK DELIMITER.
240 DO 250 I=1,J
     M=M+1
250 TEXT(M)=LINE(I)
     DO 260 I=1,6
     M=M+1
260 TEXT(M)=FORM(I)
     GO TO 160
C-----BLANK LINE HAS BEEN READ TERMINATING VALID MESSAGE.  REMOVE
C      LAST FORMAT CONTROL IN TEXT.
270 DO 280 I=1,6
     TEXT(M)=IBLNK
280 M=M-1
C-----CHECK FOR CODES IN MESSAGE TEXT.  0 TERMINATES THE RUN, AND
C      CODES 1 AND 2 PRODUCE BLANK (NULL) MESSAGES.
     IF(TEXT(1).EQ.ZER) GO TO 110
     IF(TEXT(1).EQ.ONE) GO TO 300
     IF(TEXT(1).EQ.TWO) GO TO 300
C-----MESSAGE IS EITHER UNCODED, OR CODED AS 3.  IF MESSAGE IS
C      UNCODED, PACK 1 TO M CHARACTERS FROM TEXT INTO WORDS 1 TO J
C      OF ARRAY PACK.  IF MESSAGE IS CODED AS 3, SET PACK(1) TO CODE
C      THEN PACK 2 TO M CHARACTERS FROM TEXT INTO WORDS 2 TO J OF
C      PACK.  MOVE TO AUDIT AND FILE UPDATE WHEN DONE.
     IF(TEXT(1).EQ.THR) GO TO 290
     IO=1
     LEN=M
     J=LEN+9

```

```

      J=J/10
      GO TO 295
290  IO=2
      LEN=M-1
      J=LEN+9
      J=J/10
      J=J+1
      PACK(1)=TEXT(1)
      IF(M.GT.1) GO TO 295
      WRITE(LP,1110)
      IF(ISW.EQ.1) STOP
      GO TO 130
295  ENCODE(LEN,1130,PACK(10)) (TEXT(I),I=10,M)
C-----PACKED MESSAGE LENGTH CANNOT BE LESS THAN MINIMUM RECORD LENGTH
      IF(J.GE.3) GO TO 310
      WRITE(LP,1135)
      IF(ISW.EQ.1) STOP
      GO TO 130
C-----PREPARE OUTPUT FOR MESSAGES CODED AS 1 OR 2. CODE WORD IS
C      FOLLOWED BY TWO BLANK WORDS FOR MINIMUM MESSAGE LENGTH OF 3
C      WORDS.
300  J=3
      PACK(1)=TEXT(1)
      PACK(2)=IBLNK
      PACK(3)=IBLNK
C-----DISPLAY MESSAGE FROM PACKED, CODED FORMAT
310  IF(PACK(1).EQ.ONE) GO TO 320
      IF(PACK(1).EQ.TWO) GO TO 320
      IF(PACK(1).EQ.THR) GO TO 320
      WRITE(LP,1140)
      JP=J+1
      PACK(JP)=TERM
      WRITE(LP,FPCK)
      GO TO 330
320  WRITE(LP,1150) PACK(1)
      IF(PACK(1).NE.THR) GO TO 330
      JP=J+1
      PACK(JP)=TERM
      PACK(1)=FPCK(1)
      WRITE(LP,PACK)
      PACK(1)=THR
C-----AFTER AUDIT, IN INTERACTIVE MODE, QUERY USER FOR VERIFICATION.
C      ANY NON-BLANK RESPONSE CANCELS MESSAGE.
330  IF(ISW.EQ.1) GO TO 350
      WRITE(LP,1160)
      BUFF=IBLNK
      IF(EOF(IN)) 340,340
340  READ(IN,1170) BUFF
      IF(BUFF.NE.IBLNK) GO TO 130
C-----WRITE MESSAGE TO OUTPUT FILE, THEN REPEAT ENTIRE PROCESS
C      FOR NEXT MESSAGE, UP TO 256 MESSAGES.
350  CALL WRITMS(UTXT,PACK,J,ISTR)
      GO TO 100
C
10000FORMAT (//40H  HACS DATA FIELD MESSAGE UPDATE PROGRAM//47H  ENTER
1010  FORMAT (I1)
10200FORMAT (5X,54H*****ERROR - DFAULT FILE DOES NOT CONTAIN 256 ENTRI
1030  FORMAT (//50H  ENTER NUMBER OF LAST MESSAGE SAVED IN I3 FORMAT:)
1040  FORMAT (I3)
10500FORMAT (5X,54H*****ERROR - LAST MESSAGE NUMBER NOT IN RANGE 0 TO 2
10600FORMAT (5X,21H*****ERROR - MESSAGE ,I3,33H IS OUT OF SEQUENCE ON I
10700FORMAT (/5X,34HENTER 0 OR CODED TEXT FOR MESSAGE ,I3,2H, ,A4,A3,
1 7H FIELD ,I4,2X,3A4,1H:/)

```

```

1080 FORMAT (130A1)
1090 FORMAT (5X,130A1)
11000FORMAT (5X,18H*****ERROR - LINE ,I2,48H EXCEEDS LENGTH OF 70 CHARA
1CTERS. RE-ENTER LINE.)
11100FORMAT (5X,68H*****ERROR - MESSAGE NOT FOUND. ENTER TEXT FOLLOWED
1BY A BLANK LINE.)
11200FORMAT (5X,73H*****ERROR - MESSAGE EXCEEDS MAXIMUM LENGTH OF 9 LIN
1ES. RE-ENTER MESSAGE.)
1130 FORMAT (684A1)
11350FORMAT (5X,82H*****ERROR - MESSAGE TOO SHORT (MINIMUM LENGTH = 21
1CHARACTERS). RE-ENTER MESSAGE.)
1140 FORMAT (5X,24HMESSAGE IS UNCODED TEXT:/)
1150 FORMAT (5X,25HMESSAGE IS CODED AS TYPE ,A1/)
1160 FORMAT (4H OK?)
1170 FORMAT (A10)
      END
READY.

```

7.2.2 Scenario Text (File RPSTXT)

OPROGRAM TXTLOD(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
1 TAPE10,TAPE11,TAPE12)

PROGRAM LOADS A MASS STORAGE FILE WITH CODED MESSAGE TEXT DATA, INDEXED BY MESSAGE NUMBER. TWO MODES OF OPERATION ARE PROVIDED: INTERACTIVE AND BATCH. THE FILE IS INITIALLY BUILT IN THE INTERACTIVE MODE IN WHICH A SERIES OF TEXT MESSAGES ARE ADDED TO THE PREVIOUS CONTENTS OF THE MS FILE. MESSAGES ARE ADDED IN SEQUENCE BY MESSAGE NUMBER IMMEDIATELY FOLLOWING THE LAST MESSAGE PREVIOUSLY ENTERED. INTERACTIVE RUNS MAY BE TERMINATED AND RE-STARTED AT A LATER TIME UNTIL THE FULL 256 MESSAGE FILE IS CREATED. IN BATCH MODE, A TEXT FILE, PRODUCED BY OUTPUT FROM A SEPARATE PROGRAM FOR ON-LINE TEXT EDITING, CONTAINING THE FULL SET OF TEXT MESSAGES IS PROCESSED FROM START TO FINISH. IN GENERAL, ERRORS ENCOUNTERED IN THE BATCH MODE ARE IMMEDIATELY FATAL, WHILE THE PROGRAM PERMITS RECOVERY FROM MOST INTERACTIVE ERRORS.

THE HACS DEFAULT FILE IS READ BY THIS PROGRAM TO OBTAIN FIELD NUMBERS, NAMES AND VARIABLE TYPES FOR USE IN DISPLAYS AND TO CORRELATE THE SEQUENTIAL MESSAGE NUMBERS (FROM 1 TO 256) WITH FIELD POSITIONS IN THE DEFAULT FILE.

THE MASS STORAGE FILE PRODUCED BY THIS PROGRAM CONTAINS VARIABLE LENGTH RECORDS, INDEXED NUMERICALLY BY A MESSAGE NUMBER WHICH RANGES FROM 1 TO 256. A SINGLE RECORD IS CREATED IN THE FILE FOR EACH MESSAGE NUMBER. RECORDS IN THE FILE ARE CODED AS FOLLOWS:

- UNCODED - UNCODED RECORDS CONTAIN TEXT (SEE BELOW)
- TYPE 1 - TYPE 1 RECORDS CONTAIN 3 WORDS OF 10 CHARACTERS TO CONSTRUCT THE MINIMUM RECORD SIZE FOR A CDC MS FILE. THE FIRST WORD CONTAINS 1H1, FOLLOWED BY BLANKS; THE SECOND AND THIRD WORDS ARE ALSO BLANK. TYPE 1 MESSAGES ARE STANDARDIZED IN HACS AS REFERENCES TO THE USER MANUAL.
- TYPE 2 - THESE ARE SIMILAR TO TYPE 1 RECORDS, BUT CONTAIN 1H2 FOLLOWED BY BLANKS. TYPE 2 MESSAGES ARE STANDARDIZED IN HACS AS REFERENCES TO CHEMICAL PROPERTY DATA.
- TYPE 3 - THESE ARE SPECIAL TYPE 2 RECORDS FOR WHICH ADDITIONAL TEXT IS ALSO GIVEN. THE FIRST WORD OF THE RECORD CONTAINS 1H3 FOLLOWED BY BLANKS. THE REMAINDER OF THE RECORD CONTAINS VARIABLE LENGTH TEXT.

TEXT CONTAINED IN UNCODED OR TYPE 3 RECORDS IS VARIABLE LENGTH (RECORD LENGTH GE 3), AND PACKED AS 10 CHARACTERS PER WORD. RECORDS CONTAIN ONE OR MORE LINES OF MESSAGE TEXT ORIGINALLY ENTERED VIA THIS PROGRAM IN INTERACTIVE MODE. EMBEDDED BLANKS BETWEEN LINES ARE AUTOMATICALLY REMOVED, AND FORTRAN FORMAT CODE FOR '/5X,' IS AUTOMATICALLY INSERTED BETWEEN CHARACTER STRINGS REPRESENTING DIFFERENT LINES. WHEN USED BY HACS, THE MESSAGE DATA IS READ INTO AN ARRAY AND FORMAT CODE APPENDED TO THE BEGINNING AND END OF THE ARRAY. THE EXPANDED ARRAY IS THEN USED AS AN EXECUTION TIME FORMAT TO PRODUCE THE DESIRED MESSAGE. THIS PROCEDURE IS ILLUSTRATED IN THE AUDIT PORTION OF THE PROGRAM BELOW.

SINCE THE MESSAGE TEXT IS STORED AND USED AS PACKED CHARACTER DATA, THIS PROGRAM IS SIGNIFICANTLY MACHINE DEPENDENT. OUTPUT PRODUCED BY THIS PROGRAM IS WRITTEN AS 10 CHARACTER WORDS. IN ADDITION TO THE USE OF OPEN, WRITE AND CLOSE MASS STORAGE UTILITY SUBROUTINES, THE CDC ENCODE FUNCTION IS USED TO PACK MESSAGE DATA READ IN A1 FORMAT TO A10 FORMAT FOR OUTPUT TO THE MS FILE.

BUFF = A10 STORAGE FOR USER RESPONSE TO VERIFICATION QUERY. ANY RESPONSE OTHER THAN CARRIAGE RETURN REJECTS MESSAGE INPUT.
 DEFLT = EXTERNAL HACS DEFAULT FILE, READ TO OBTAIN HACS FIELD NUMBER, FIELD NAME AND FIELD TYPE (INTEGER OR REAL) CORRESPONDING TO SEQUENTIAL FIELD MESSAGE NUMBERS.
 FORM = SIX-WORD ARRAY CONTAINING FORMAT CODE TO BE INSERTED BETWEEN EACH LINE OF MULTIPLE LINE MESSAGES.
 FPCK = ARRAY EQUIVALENCED TO PACKED MESSAGE TEXT, AND CONTAINING STANDARD FORMAT CODE IN FIRST WORD
 FVAL = ARRAY OF DEFAULT VALUES, MINIMUM AND MAXIMUM FOR REAL HACS DATA FIELDS. READ AS PART OF DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 I = GENERAL LOOP INDEX
 IO = FIRST CHARACTER OF ARRAY TEXT FOR START OF PACK INTO ARRAY PACK, SET TO 1 FOR UNCODED MESSAGES OR TO 2 FOR TYPE 3 MESSAGES.
 IBLNK = DATA WORD SET TO ALL BLANKS FOR USE IN INITIALIZING CHARACTER VARIABLES
 IBTCH = DEVICE NUMBER FOR BATCH INPUT FILE WHEN RUNNING IN BATCH MODE
 IN = INPUT DEVICE NUMBER SET TO EITHER TTY OR IBTCH VIA USER CONTROL AT START OF RUN
 ISTRT = INDEX NUMBER OF LAST MESSAGE PREVIOUSLY LOADED ONTO MASS STORAGE FILE. EACH RUN OF PROGRAM IN THE INTERACTIVE MODE ADDS MESSAGES SEQUENTIALLY BY 1 FROM ISTRT+1 UP TO A MAXIMUM OF 256. IN BATCH MODE, A BULK RE-CREATION OF THE MS FILE BUILDS ALL MESSAGES FROM 1 TO 256.
 ISW = CONTROL FLAG SET TO 0 FOR INTERACTIVE INPUT, OR TO 1 FOR BATCH INPUT.
 IVAL = ARRAY OF DEFAULT VALUES, MINIMUM AND MAXIMUM FOR INTEGER HACS DATA FIELDS. READ AS PART OF DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 IVAR = CODED FIELD TYPE INDICATOR PACKED IN ARRAY LIST FOR EACH HACS DATA FIELD (0=INTEGER FIELD, 1=REAL FIELD)
 J = GENERAL LOOP INDEX
 = INDEX TO LAST NON-BLANK CHARACTER READ ON SINGLE MESSAGE LINE
 = WORD INDEX TO ARRAY PACK, AND LENGTH OF FINAL PACKED MESSAGE TEXT (IN WORDS)
 JP = DUMMY INDEX, SET TO J+1, USED TO STORE FORMAT TERMINATOR IN PACKED MESSAGE TEXT FOR DISPLAY BUT NOT OUTPUT TO MS FILE.
 LEN = COUNT OF CHARACTERS IN TEXT TO BE MOVED TO ARRAY PACK IN A10 FORMAT
 LINE = BUFFER USED FOR STORAGE OF SINGLE INPUT LINE OF MESSAGE TEXT IN A1 FORMAT. LENGTH SET TO 130 CHARACTERS FOR USE WITH WIDE-BODY TERMINALS, HOWEVER ACTUAL MESSAGE LENGTH IS LIMITED TO 70 CHARACTERS PER LINE.
 LIST = ARRAY OF CODES IN HACS DEFAULT FILE DEFINING STRUCTURE OF DATA FIELD ITEM I AS LIST(I,J) WHERE J=1,6. LIST(I,1) GIVES FIELD NUMBER FOR FIELD I AND LIST(I,J), J=3,5, GIVES FIELD NAME. ELEMENT 2 CONTAINS CODED SOURCE CODE, VARIABLE TYPE AND QUANTITY TYPE. ELEMENT 6 IS INDEX TO STORAGE OF NUMERIC VALUES IN FVAL OR IVAL. REFER TO HACS PROGRAM DOCUMENTATION FOR COMPLETE DETAILS.
 LP = UNIT DEVICE FOR PROGRAM OUTPUT IN EITHER BATCH OR INTERACTIVE MODES
 M = COUNT OF TOTAL NUMBER OF MESSAGE CHARACTERS ACCUMULATED IN ARRAY TEXT
 MNF = MAXIMUM NUMBER OF REAL FIELD ITEMS ALLOWED IN HACS DEFAULT FILE. READ FROM DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.
 MNI = MAXIMUM NUMBER OF INTEGER FIELD ITEMS ALLOWED IN HACS DEFAULT FILE. READ FROM DEFAULT FILE BUT NOT OTHERWISE USED IN THIS PROGRAM.

MOD = DATA WORD CONTAINING LABELS USED IN MESSAGE INPUT
 MSG = PROMPT DISPLAY FOR REAL OR INTEGER FIELDS.
 NDEX = TEXT LABEL STORED AS PART OF DEFAULT FILE
 = FILE TABLE USED BY MASS STORAGE ROUTINES, DIMENSIONED
 TO NUMBER OF TEXT MESSAGES + 1
 NF = ACTUAL NUMBER OF REAL DATA FIELDS CONTAINED IN HACs
 DEFAULT FILE
 NFLD = TOTAL NUMBER OF DATA FIELDS STORED IN HACs DEFAULT
 FILE (=NF+NI), DEFINITIONS OF EACH MESSAGE ARE
 STORED SEQUENTIALLY IN ARRAY LIST.
 NI = ACTUAL NUMBER OF INTEGER DATA FIELDS CONTAINED IN HACs
 DEFAULT FILE
 NLIN = LINE COUNTER, USED TO LIMIT MULTIPLE LINE TEXT
 MESSAGES TO NOT MORE THAN 9 LINES OF TEXT FOLLOWED
 BY SINGLE BLANK LINE AS A DELIMITER.
 ONE = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 1
 PACK = ARRAY CONTAINING PACKED MESSAGE TEXT DATA TO BE
 WRITTEN TO MS FILE.
 TERM = DATA WORD CONTAINING CHARACTERS TO CLOSE MESSAGE
 IN PACK FOR USE AS EXECUTION TIME FORMAT
 TEXT = ARRAY USED TO ACCUMULATE MESSAGE LINE INPUT AND
 FORMAT CONTROL LINE SEPARATION CHARACTERS IN
 SINGLE CHARACTER (A1) FORMAT. ALL EMBEDDED BLANKS
 BETWEEN LINES ARE REMOVED BEFORE PACKING.
 THR = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 3
 TTY = DEVICE NUMBER USED FOR INPUT UNIT WHEN RUNNING IN
 INTERACTIVE MODE
 TWO = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 2
 UDFLT = UNIT DEVICE NUMBER FOR EXTERNAL HACs DEFAULT FILE
 UTXT = UNIT DEVICE NUMBER FOR EXTERNAL MS MESSAGE TEXT FILE
 ZER = DATA WORD CONTAINING TAG USED TO TEST FOR USER
 TERMINATION OF INTERACTIVE TERMINAL SESSION.

AUTHOR - R.G. POTTS
 ARTHUR D. LITTLE, INC.
 35/318A ACORN PARK
 CAMBRIDGE, MASS. 02140
 TEL. 617-864-5770, EXT. 2813

DATE - 22 JULY 1980

OCOMMON/BASE/MSG(10),MNF,MNI,NF,NI,LIST(275,6),
 1 FVAL(225,3),IVAL(50,3)

REAL MSG
 DIMENSION DEFLT(2489)
 EQUIVALENCE (DEFLT(1),MSG(1))

ODIMENSION FORM(6),FPCK(71),LINE(130),MOD(2,2),NDEX(257),
 1 PACK(70),TEXT(684)
 OINTEGER BUFF,FORM,FPCK,ONE,PACK,TERM,TEXT,THR,TTY,TWO,
 1 UDFLT,UTXT,ZER
 EQUIVALENCE (FPCK(2),PACK(1))
 DATA (FORM(I),I=1,6)/1H",1H/,1H9,1HX,1H,,1H"/
 DATA FPCK(1)/10H(9X,"/,1RLNK/1H/,1RTCH/12/,LP/6/
 ODATA MOD(1,1)/4HINTE/,MOD(1,2)/3HGER/,MOD(2,1)/4HREAL/,
 1 MOD(2,2)/1H/,ONE/1H1/,TERM/2H"/,THR/1H3/,TTY/5/,
 2 TWO/1H2/,UDFLT/10/,UTXT/11/,ZER/1H0/

-----SELECT BATCH OR INTERACTIVE INPUT UNIT

10 WRITE(TTY,1000)
 IF(EOF(TTY)) 20,20
 20 READ(TTY,1010) ISW
 IF(ISW.EQ.0) GO TO 30
 IF(ISW.NE.1) GO TO 10
 IN=IBTCH
 GO TO 40
 30 IN=TTY
 40 CONTINUE

```

C-----READ EXTERNAL DEFAULT FILE CONTAINING FIELD NUMBERS AND
C      NAMES INDEXED SEQUENTIALLY.
      REWIND UDFLT
      READ(UDFLT) DEFLT
C
C-----PROGRAM ASSUMES DEFAULT FILE CONTAINS EXACTLY 256 ENTRIES.
      NFLD=NF+NI
      IF(NFLD.EQ.256) GO TO 50
      WRITE(LP,1020)
      STOP
      50 CONTINUE
C
C-----OPEN MASS STORAGE MESSAGE FILE, INDEXED BY MESSAGE NUMBER
      CALL OPENMS(UTXT,NDEX,32,0)
C
C-----READ AND VALIDATE NUMBER OF LAST MESSAGE ENTERED TO INITIALIZE
C      START OF SEQUENCE.
      60 IF(ISW.EQ.0) WRITE(TTY,1030)
      IF(EOF(IN)) 70,70
      70 READ(IN,1040) ISTRT
      IF(ISTRT.LT.0) GO TO 80
      IF(ISTRT.LE.31) GO TO 90
      80 WRITE(LP,1050)
      IF(ISW.EQ.0) GO TO 60
      STOP
      90 CONTINUE
C
C-----START OF EACH NEW CYCLE FOR NEXT MESSAGE. CLOSE FILE AND STOP
C      AFTER 256 MESSAGES ARE STORED.
      100 ISTRT=ISTRT+1
      IF(ISTRT.LE.31) GO TO 120
      110 CALL CLOSMS(UTXT)
      STOP
      120 CONTINUE
C
C-----READ SINGLE LINE FROM BATCH INPUT FILE (SKIP FOR INTERACTIVE
C      INPUT). LINE IN FILE GIVES MESSAGE NUMBER, HACS FIELD NUMBER
C      AND HACS FIELD NAME FOR REFERENCE USE. ONLY THE MESSAGE
C      NUMBER IS READ AND MUST MATCH THE SEQUENCE COUNT ISTRT.
      IF(ISW.EQ.0) GO TO 130
      READ(IN,1040) NMSG
      IF(NMSG.EQ.ISTRT) GO TO 130
      WRITE(LP,1060) NMSG
      STOP
C
C-----WRITE PROMPT DISPLAY FOR EITHER INTERACTIVE OR BATCH INPUT
C      GIVING THE MESSAGE NUMBER, FIELD MODE, NUMBER AND NAME.
      130 IVAR=LIST(ISTRT,2)/1000
      IVAR=IVAR+1
      OWRITE(LP,1070) ISTRT,(MOD(IVAR,I),I=1,2),LIST(ISTRT,1),
      1 (LIST(ISTRT,J),J=3,5)
C
C-----INITIALIZE FOR ACCUMULATION OF MULTIPLE LINE MESSAGES
      M=0
      DO 140 I=1,684
      140 TEXT(I)=IBLNK
      DO 150 I=1,70
      150 PACK(I)=IBLNK
      NLIN=0
C
C-----RETURN HERE TO READ EACH NEW LINE OF MULTIPLE LINE MESSAGE,
C      OR BLANK LINE TERMINATING MESSAGE. INITIALIZE.
      160 NLIN=NLIN+1
      170 DO 180 I=1,130
      180 LINE(I)=IBLNK
C
C-----READ INPUT LINE OF UP TO 130 CHARACTERS IN A1 FORMAT. THE
C      FORMAT FOR INTERACTIVE INPUT SUPPRESSES LEADING SPACES. FOR
C      BATCH INPUT, THE TEXT IS PRECEDED BY FIVE SPACES SINCE THE
C      BATCH FILE CREATED FOR EDITING IS PREPARED BY EXECUTING THE
C      PREVIOUS MESSAGES AS FORMAT STATEMENTS.
      IF(EOF(IN)) 190,190

```

```

190 CONTINUE
    IF(ISW.EQ.0) READ(IN,1080) LINE
    IF(ISW.EQ.1) READ(IN,1090) LINE
C-----LOCATE THE LAST NON-BLANK CHARACTER READ ON THE INPUT LINE.
    DO 200 I=1,130
        J=131-I
        IF(LINE(J).NE.IBLNK) GO TO 210
    200 CONTINUE
        J=0
    210 CONTINUE
C-----J GIVES THE INDEX TO THE LAST NON-BLANK CHARACTER READ ON
C-----INPUT.  VALID TEXT LINES MAY BE BLANK IF NOT THE FIRST LINE,
C-----AND MAY NOT CONTAIN MORE THAN 70 CHARACTERS OF TEXT.
    IF(J.LE.70) GO TO 220
C-----SINGLE LINE OVERFLOW.
    WRITE(LP,1100) NLIN
    IF(ISW.EQ.1) STOP
    GO TO 170
C-----TEST FOR LAST LINE (BLANK) TERMINATING MESSAGE.
    220 IF(J.GT.0) GO TO 230
C-----IF BLANK LINE IS PRECEDED BY TEXT, HAVE VALID END OF MESSAGE.
C-----BRANCH TO PROCESS.  OTHERWISE, HAVE ERROR.
    IF(M.GT.0) GO TO 270
    WRITE(LP,1110)
    IF(ISW.EQ.1) STOP
    GO TO 170
C-----GET HERE WITH MESSAGE TEXT ENTERED FOR SINGLE LINE.  TEXT
C-----MESSAGES ARE ALLOWED FOR UP TO 9 NON-BLANK LINES OF TEXT.  AN
C-----ERROR OCCURS IF MORE THAN 9 LINES ARE ENTERED.
    230 IF(NLIN.LE.9) GO TO 240
    WRITE(LP,1120)
    IF(ISW.EQ.1) STOP
    GO TO 130
C-----APPEND NEW LINE OF TEXT TO ACCUMULATED MESSAGE TEXT, INSERT
C-----OUTPUT FORMAT CONTROL AT END OF EACH NEW LINE, THEN RETURN TO
C-----READ NEXT TEXT LINE OR BLANK DELIMITER.
    240 DO 250 I=1,J
        M=M+1
    250 TEXT(M)=LINE(I)
        DO 260 I=1,6
            M=M+1
    260 TEXT(M)=FORM(I)
        GO TO 160
C-----BLANK LINE HAS BEEN READ TERMINATING VALID MESSAGE.  REMOVE
C-----LAST FORMAT CONTROL IN TEXT.
    270 DO 280 I=1,6
        TEXT(M)=IBLNK
    280 M=M-1
C-----CHECK FOR CODES IN MESSAGE TEXT.  0 TERMINATES THE RUN, AND
C-----CODES 1 AND 2 PRODUCE BLANK (NULL) MESSAGES.
    IF(TEXT(1).EQ.ZER) GO TO 110
    IF(TEXT(1).EQ.ONE) GO TO 300
    IF(TEXT(1).EQ.TWO) GO TO 300
C-----MESSAGE IS EITHER UNCODED, OR CODED AS 3.  IF MESSAGE IS
C-----UNCODED, PACK 1 TO M CHARACTERS FROM TEXT INTO WORDS 1 TO J
C-----OF ARRAY PACK.  IF MESSAGE IS CODED AS 3, SET PACK(1) TO CODE
C-----THEN PACK 2 TO M CHARACTERS FROM TEXT INTO WORDS 2 TO J OF
C-----PACK.  MOVE TO AUDIT AND FILE UPDATE WHEN DONE.
    IF(TEXT(1).EQ.THR) GO TO 290
    IO=1
    LEN=M
    J=LEN+9

```

```

        J=J/10
        GO TO 295
290   IO=2
        LEN=M-1
        J=LEN+9
        J=J/10
        J=J+1
        PACK(1)=TEXT(1)
        IF(M.GT.1) GO TO 295
        WRITE(LP,1110)
        IF(ISW.EQ.1) STOP
        GO TO 130
295   ENCODE(LEN,1130,PACK(IO)) (TEXT(I),I=IO,M)
C-----
C-----PACKED MESSAGE LENGTH CANNOT BE LESS THAN MINIMUM RECORD LENGTH
        IF(J.GE.3) GO TO 310
        WRITE(LP,1135)
        IF(ISW.EQ.1) STOP
        GO TO 130
C-----
C-----PREPARE OUTPUT FOR MESSAGES CODED AS 1 OR 2. CODE WORD IS
C-----FOLLOWED BY TWO BLANK WORDS FOR MINIMUM MESSAGE LENGTH OF 3
C-----WORDS.
300   J=3
        PACK(1)=TEXT(1)
        PACK(2)=IBLNK
        PACK(3)=IBLNK
C-----
C-----DISPLAY MESSAGE FROM PACKED, CODED FORMAT
310   IF(PACK(1).EQ.ONE) GO TO 320
        IF(PACK(1).EQ.TWO) GO TO 320
        IF(PACK(1).EQ.THR) GO TO 320
        WRITE(LP,1140)
        JP=J+1
        PACK(JP)=TERM
        WRITE(LP,FPCK)
        GO TO 330
320   WRITE(LP,1150) PACK(1)
        IF(PACK(1).NE.THR) GO TO 330
        JP=J+1
        PACK(JP)=TERM
        PACK(1)=FPCK(1)
        WRITE(LP,PACK)
        PACK(1)=THR
C-----
C-----AFTER AUDIT, IN INTERACTIVE MODE, QUERY USER FOR VERIFICATION.
C-----ANY NON-BLANK RESPONSE CANCELS MESSAGE.
330   IF(ISW.EQ.1) GO TO 350
        WRITE(LP,1160)
        BUFF=IBLNK
        IF(EOF(IN)) 340,340
340   READ(IN,1170) BUFF
        IF(BUFF.NE.IBLNK) GO TO 130
C-----
C-----WRITE MESSAGE TO OUTPUT FILE, THEN REPEAT ENTIRE PROCESS
C-----FOR NEXT MESSAGE, UP TO 256 MESSAGES.
350   CALL WRITMS(UTXT,PACK,J,ISTRT)
        GO TO 100
C-----
10000FORMAT (//40H  HACS DATA FIELD MESSAGE UPDATE PROGRAM//47H  ENTER
1010 10 FOR INTERACTIVE OR 1 FOR BATCH INPUT:)
1010 FORMAT (I1)
10200FORMAT (5X,54H*****ERROR - DEFAULT FILE DOES NOT CONTAIN 256 ENTRI
1030 1ES)
1030 FORMAT (//50H  ENTER NUMBER OF LAST MESSAGE SAVED IN I3 FORMAT:)
1040 FORMAT (I3)
10500FORMAT (5X,54H*****ERROR - LAST MESSAGE NUMBER NOT IN RANGE 0 TO 2
1060 156)
10600FORMAT (5X,21H*****ERROR - MESSAGE ,I3,33H IS OUT OF SEQUENCE ON I
1070 1INPUT FILE)
10700FORMAT (/5X,34HENTER 0 OR CODED TEXT FOR MESSAGE ,I3,2H, ,A4,A3,
1 7H FIELD ,I4,2X,3A4,1H:/)

```

```

1080 FORMAT (130A1)
1090 FORMAT (9X,130A1)
11000FORMAT (5X,18H*****ERROR - LINE ,12,48H EXCEEDS LENGTH OF 70 CHARA
1CTERS. RE-ENTER LINE.)
11100FORMAT (5X,68H*****ERROR - MESSAGE NOT FOUND. ENTER TEXT FOLLOWED
1BY A BLANK LINE.)
11200FORMAT (5X,73H*****ERROR - MESSAGE EXCEEDS MAXIMUM LENGTH OF 9 LIN
1ES. RE-ENTER MESSAGE.)
1130 FORMAT (684A1)
11350FORMAT (5X,82H*****ERROR - MESSAGE TOO SHORT (MINIMUM LENGTH = 21
1CHARACTERS). RE-ENTER MESSAGE.)
1140 FORMAT (5X,24HMESSAGE IS UNCODED TEXT:/)
1150 FORMAT (5X,25HMESSAGE IS CODED AS TYPE ,A1/)
1160 FORMAT (4H OK?)
1170 FORMAT (A10)
      END
READY.

```

7.2.3 Model Text (File MTXLOD)

PROGRAM MTXLOD(OUTPUT,TAPE6=OUTPUT,TAPE11,TAPE12)

C
C
C
C
C
C
C
C
C

PROGRAM MTXLOD (FOR MESSAGE TEXT LOAD) WAS CREATED BY ADAPTING THE FIELD TEXT LOAD PROGRAM FOR THE FOLLOWING SPECIAL CASE. ONLY BATCH INPUT IS ALLOWED FROM A FILE OF UNCODED MESSAGE TEXT DATA. EACH MESSAGE IS SEPARATED BY A BLANK LINE, CONTAINS AT LEAST ONE LINE OF TEXT, AND A MAXIMUM OF 25 LINES OF TEXT. EACH LINE IS LIMITED TO NOT MORE THAN 70 CHARACTERS. THE COMPLETE FILE CONTAINS EXACTLY 29 MESSAGES. REFER TO LISTING OF PROGRAM TXTLOD FOR DEFINITIONS AND ADDITIONAL INFORMATION.

C
C

```

DIMENSION FORM(6),FPCK(192),LINE(80),NDEX(30),PACK(191),
1 TEXT(1900)
INTEGER FORM,FPCK,PACK,TERM,TEXT,UTXT
EQUIVALENCE (FPCK(2),PACK(1))
0DATA (FORM(I),I=1,6)/1H*,1H/,1H9,1HX,1H,,1H*/,
1 FPCK(1)/10H(9X,*/,IBLNK/1H /,IN/11/,LP/6/,
2 TERM/2H*)/,UTXT/12/

MSG=0
CALL OPENMS(UTXT,NDEX,30,0)
100 MSG=MSG+1
IF(MSG.LE.29) GO TO 120
CALL CLOSMS(UTXT)
STOP
120 M=0
DO 140 I=1,1900
140 TEXT(I)=IBLNK
DO 150 I=1,190
150 PACK(I)=IBLNK
NLIN=0
160 NLIN=NLIN+1
DO 180 I=1,80
180 LINE(I)=IBLNK
READ(IN,1090) LINE
DO 200 I=1,80
J=81-I
IF(LINE(J).NE.IBLNK) GO TO 210
200 CONTINUE
J=0
210 CONTINUE
IF(J.LE.70) GO TO 220
WRITE(LP,1100) LINE
STOP
220 IF(J.GT.0) GO TO 230
IF(M.GT.0) GO TO 270
WRITE(LP,1110)
STOP
230 IF(NLIN.LE.25) GO TO 240
WRITE(LP,1120)
STOP
240 DO 250 I=1,J
M=M+1
250 TEXT(M)=LINE(I)
DO 260 I=1,6
M=M+1
260 TEXT(M)=FORM(I)
GO TO 160
270 M=M-6
ENCODE(M,1130,PACK(1)) (TEXT(I),I=1,M)
WRITE(LP,1140) MSG
J=M+9
J=J/10
I=J+1
PACK(I)=TERM

```

```
WRITE(LP,FPCK)  
CALL WRITMS(UTXT,PACK,J,NMSG)  
GO TO 100
```

```
C  
1090 FORMAT (80A1)  
1100 FORMAT (15H LINE OVERFLOW:,80A1)  
1110 FORMAT (14H BLANK MESSAGE)  
1120 FORMAT (15H TOO MANY LINES)  
1130 FORMAT (1900A1)  
1140 FORMAT (//1X,8HMESSAGE ,I2//)  
END
```

```
READY.
```

7.3 Message File Display

Section 7.3 contains listings of the programs used to obtain displays of the message text files created by the programs listed in Section 7.2. Three versions are provided, one for field text, one for scenario text and one for model text.

The original version of the program was written to process coded field text messages and to provide as output a file of message data. The output file can either be printed for display or edited and re-entered into the file build program for updating in batch mode. Complete details are contained in comments in the program listing.

The program used to display scenario messages is nearly, but not quite, identical to that used for field text messages. Displays appropriate for data field items, but not scenarios, were not changed.

The file containing model text data is displayed using a very simplified version of the program, with output directly to the terminal.

Note that the maximum message lengths, number of messages, format control characters, and related message attributes are different among these three programs and the files they process.

7.3.1 Field Text (File RGPMMSG)

PROGRAM DISPLAY(OUTPUT,TAPE4=OUTPUT,TAPE10,TAPE11,TAPE12)

PROGRAM WRITES A DISPLAY OF THE HACS FIELD TEXT MESSAGE FILE (TAPE 11) TO AN OUTPUT FILE (TAPE 12) IN BATCH UPDATE FORMAT. THE CONTENTS OF THE OUTPUT FILE CAN THEN BE PRINTED USING A SYSTEM UTILITY, OR CAN BE EDITED (ALSO BY A SYSTEM UTILITY). THE OUTPUT FILE IS FORMATTED SO THAT AFTER EDITING IT CAN BE PROCESSED BY THE MESSAGE UPDATE PROGRAM IN BATCH MODE TO PRODUCE AN UPDATED MESSAGE FILE. REFER TO THE MESSAGE UPDATE PROGRAM LISTING FOR A DESCRIPTION OF THE MESSAGE CODES AND MESSAGE FILE FORMAT.

THE HACS DEFAULT FILE IS READ BY THIS PROGRAM TO OBTAIN FIELD NUMBERS, NAMES AND VARIABLE TYPES FOR USE IN DISPLAYS AND TO CORRELATE THE SEQUENTIAL MESSAGE NUMBERS (FROM 1 TO 256) WITH FIELD POSITIONS IN THE DEFAULT FILE. IF THE MESSAGE FILE IS NOT FULL, THAT IS, CONTAINS LESS THAN 256 MESSAGES, THIS PROGRAM WILL TERMINATE WITH AN MS FILE READ ERROR.

CODED MESSAGES WRITTEN BY THIS PROGRAM ARE GENERATED BY READING TEXT FROM THE MESSAGE FILE, APPENDING APPROPRIATE FORTRAN FORMAT CODES TO THE MESSAGE STORED AS AN ARRAY, THEN USING THE ARRAY AS AN EXECUTION TIME FORMAT.

DEFLT = EXTERNAL HACS DEFAULT FILE, READ TO OBTAIN HACS FIELD NUMBER, FIELD NAME AND FIELD TYPE (INTEGER OR REAL) CORRESPONDING TO SEQUENTIAL FIELD MESSAGE NUMBERS.

FPCK = ARRAY EQUIVALENCED TO UNCODED, PACKED MESSAGE TEXT AND CONTAINING STANDARD FORMAT CODE IN FIRST WORD

FVAL = ARRAY OF DEFAULT VALUES FOR REAL FIELDS, READ FROM DEFAULT FILE BUT NOT USED IN THIS PROGRAM

I = INDEX ON MESSAGE NUMBER FROM 1 TO NFLD

IVAL = ARRAY OF DEFAULT VALUES FOR INTEGER FIELDS, READ FROM DEFAULT FILE BUT NOT USED IN THIS PROGRAM

IVAR = CODED FIELD TYPE INDICATOR PACKED IN ARRAY LIST FOR EACH HACS DATA FIELD (0=INTEGER FIELD, 1=REAL FIELD)

J = GENERAL SUBSCRIPT INDEX

LIST = ARRAY OF CODES IN HACS DEFAULT FILE DEFINING THE STRUCTURE OF DATA FIELD ITEM I AS LIST(I,J) WHERE J=1,6. REFER TO HACS PROGRAM DOCUMENTATION FOR COMPLETE DETAILS.

MNF = MAXIMUM NUMBER OF REAL FIELD ITEMS ALLOWED IN HACS DEFAULT FILE

MNI = MAXIMUM NUMBER OF INTEGER FIELD ITEMS ALLOWED IN HACS DEFAULT FILE

MOD = DATA ARRAY USED TO DISPLAY FIELD TYPE LABELS

MSG = TEXT LABEL STORED AS PART OF DEFAULT FILE

NDEX = FILE TABLE USED BY MASS STORAGE ROUTINES, DIMENSIONED TO NUMBER OF TEXT MESSAGES + 1

NF = ACTUAL NUMBER OF REAL DATA ITEMS CONTAINED IN HACS DEFAULT FILE

NFLD = TOTAL NUMBER OF DATA FIELDS STORED IN HACS DEFAULT FILE (=NF+NI). DEFINITIONS OF EACH MESSAGE ARE STORED SEQUENTIALLY IN ARRAY LIST.

NI = ACTUAL NUMBER OF INTEGER DATA ITEMS CONTAINED IN HACS DEFAULT FILE

NW = LENGTH OF LAST RECORD, IN WORDS, READ FROM MESSAGE FILE

ONE = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 1

OUT = UNIT DEVICE NUMBER FOR OUTPUT FILE CREATED BY THIS PROGRAM

PACK = ARRAY CONTAINING PACKED MESSAGE TEXT DATA READ FROM MS FILE, MAXIMUM RECORD LENGTH OF 69. DIMENSIONED AS 70 WORD ARRAY TO ALLOW FORMAT CODE TO BE ADDED AFTER TEXT.

```

C      STRT  = SPECIAL FORMAT TAG FOR TYPE 3 MESSAGES, CREATES
C              DISPLAY IN UPDATE FORMAT WITH CODE 3 IN FIRST
C              POSITION, FIRST CHARACTER OF MESSAGE TEXT IN
C              SECOND POSITION
C      TERM  = DATA WORD CONTAINING CHARACTERS TO CLOSE MESSAGE
C              IN PACK FOR USE AS EXECUTION TIME FORMAT.
C      THR   = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 3
C      TWO   = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 2
C      UDFLT = UNIT DEVICE NUMBER FOR EXTERNAL HACS DEFAULT FILE
C      UTXT  = UNIT DEVICE NUMBER FOR EXTERNAL MS MESSAGE TEXT FILE
C
C      AUTHOR - R.G. POTTS
C              ARTHUR D. LITTLE, INC.
C              35/318A ACORN PARK
C              CAMBRIDGE, MASS. 02140
C              TEL. 617-864-5770, EXT. 2813
C
C      DATE  - 20 AUGUST 1980
C
C      OCOMMON/BASE/MSG(10),MNF,MNI,NF,NI,LIST(275,6),
1      FVAL(225,3),IVAL(50,3)
C      REAL   MSG
C      DIMENSION DEFLT(2489)
C      EQUIVALENCE (DEFLT(1),MSG(1))
C
C      DIMENSION FPCK(71),MOD(2,2),NDEX(257),PACK(70)
C      INTEGER  FPCK,ONE,OUT,PACK,STRT,TERM,THR,TWO,UDFLT,UTXT
C      EQUIVALENCE (FPCK(2),PACK(1))
C      ODATA    FPCK(1)/10H(5X,      '/',MOD(1,1)/4HINTE/,
1      MOD(1,2)/3HGER/,MOD(2,1)/4HREAL/,MOD(2,2)/1H /,
2      ONE/1H1/,OUT/12/,STRT/10H(5X,  '3',TERM/2H*)/,
3      THR/1H3/,TWO/1H2/,UDFLT/10/,UTXT/11/
C
C      -----READ EXTERNAL DEFAULT FILE CONTAINING FIELD NUMBERS AND
C      NAMES INDEXED SEQUENTIALLY. TOTAL NUMBER OF FIELDS ON FILE
C      IS SUM OF REAL AND INTEGER FIELDS.
C      REWIND UDFLT
C      READ(UDFLT) DEFLT
C      NFLD=NF+NI
C
C      -----WRITE INITIAL MESSAGE WITH STARTING MESSAGE NUMBER FOR
C      USE IN BATCH UPDATE
C      WRITE(OUT,1000)
C
C      -----OPEN MASS STORAGE MESSAGE FILE, INDEXED BY MESSAGE NUMBER,
C      AND START LOOP ON MESSAGES FROM 1 TO NFLD
C      CALL OPENMS(UTXT,NDEX,257,0)
C      DO 40 I=1,NFLD
C
C      -----DISPLAY MESSAGE NUMBER TOGETHER WITH HACS FIELD MODE, NUMBER
C      AND NAME.
C      IVAR=LIST(I,2)/1000
C      IVAR=IVAR+1
C      WRITE(OUT,1010) I,(MOD(IVAR,J),J=1,2),LIST(I,1),(LIST(I,J),J=3,5)
C
C      -----READ AND DECODE MESSAGE I
C      CALL READMS(UTXT,PACK,69,I)
C      NW=LENGTH(UTXT)
C      IF(PACK(1).EQ.ONE) GO TO 10
C      IF(PACK(1).EQ.TWO) GO TO 10
C      NW=NW+1
C      PACK(NW)=TERM
C      IF(PACK(1).EQ.THR) GO TO 20
C
C      -----OUTPUT UNCODED MESSAGE
C      WRITE(OUT,FPCK)
C      GO TO 30
C
C      -----WRITE CODED MESSAGES, TYPES 1 AND 2

```

```

10 WRITE(OUT,1020) PACK(1)
   GO TO 30
C-----WRITE CODED MESSAGE, TYPE 3, IN UPDATE FORMAT
20 PACK(1)=STRT
   WRITE(OUT,PACK)
C-----WRITE MESSAGE DELIMITER, THEN CONTINUE LOOP FOR ALL MESSAGES.
30 WRITE(OUT,1030)
40 CONTINUE
   STOP
C
1000 FORMAT (42H000 = NUMBER OF LAST MESSAGE ON PRIOR FILE)
10100FORMAT (I3,21H = MESSAGE INDEX FOR ,A4,A3,7H FIELD ,I4,2X,3A4,
1 15H, MESSAGE TEXT;)
1020 FORMAT (5X,A1)
1030 FORMAT (5X)
   END
READY.

```

7.3.2 Scenario Text (File RPSMSG)

PROGRAM DISPLAY(OUTPUT,TAPE6=OUTPUT,TAPE10,TAPE11,TAPE12)

PROGRAM WRITES A DISPLAY OF THE HACS FIELD TEXT MESSAGE FILE (TAPE 11) TO AN OUTPUT FILE (TAPE 12) IN BATCH UPDATE FORMAT. THE CONTENTS OF THE OUTPUT FILE CAN THEN BE PRINTED USING A SYSTEM UTILITY, OR CAN BE EDITED (ALSO BY A SYSTEM UTILITY). THE OUTPUT FILE IS FORMATTED SO THAT AFTER EDITING IT CAN BE PROCESSED BY THE MESSAGE UPDATE PROGRAM IN BATCH MODE TO PRODUCE AN UPDATED MESSAGE FILE. REFER TO THE MESSAGE UPDATE PROGRAM LISTING FOR A DESCRIPTION OF THE MESSAGE CODES AND MESSAGE FILE FORMAT.

THE HACS DEFAULT FILE IS READ BY THIS PROGRAM TO OBTAIN FIELD NUMBERS, NAMES AND VARIABLE TYPES FOR USE IN DISPLAYS AND TO CORRELATE THE SEQUENTIAL MESSAGE NUMBERS (FROM 1 TO 256) WITH FIELD POSITIONS IN THE DEFAULT FILE. IF THE MESSAGE FILE IS NOT FULL, THAT IS, CONTAINS LESS THAN 256 MESSAGES, THIS PROGRAM WILL TERMINATE WITH AN MS FILE READ ERROR.

CODED MESSAGES WRITTEN BY THIS PROGRAM ARE GENERATED BY READING TEXT FROM THE MESSAGE FILE, APPENDING APPROPRIATE FORTRAN FORMAT CODES TO THE MESSAGE STORED AS AN ARRAY, THEN USING THE ARRAY AS AN EXECUTION TIME FORMAT.

DEFLT = EXTERNAL HACS DEFAULT FILE, READ TO OBTAIN HACS FIELD NUMBER, FIELD NAME AND FIELD TYPE (INTEGER OR REAL) CORRESPONDING TO SEQUENTIAL FIELD MESSAGE NUMBERS.

FPCK = ARRAY EQUIVALENCED TO UNCODED, PACKED MESSAGE TEXT AND CONTAINING STANDARD FORMAT CODE IN FIRST WORD

FVAL = ARRAY OF DEFAULT VALUES FOR REAL FIELDS, READ FROM DEFAULT FILE BUT NOT USED IN THIS PROGRAM

I = INDEX ON MESSAGE NUMBER FROM 1 TO NFLD

IVAL = ARRAY OF DEFAULT VALUES FOR INTEGER FIELDS, READ FROM DEFAULT FILE BUT NOT USED IN THIS PROGRAM

IVAR = CODED FIELD TYPE INDICATOR PACKED IN ARRAY LIST FOR EACH HACS DATA FIELD (0=INTEGER FIELD, 1=REAL FIELD)

J = GENERAL SUBSCRIPT INDEX

LIST = ARRAY OF CODES IN HACS DEFAULT FILE DEFINING THE STRUCTURE OF DATA FIELD ITEM I AS LIST(I,J) WHERE J=1,6. REFER TO HACS PROGRAM DOCUMENTATION FOR COMPLETE DETAILS.

MNF = MAXIMUM NUMBER OF REAL FIELD ITEMS ALLOWED IN HACS DEFAULT FILE

MNI = MAXIMUM NUMBER OF INTEGER FIELD ITEMS ALLOWED IN HACS DEFAULT FILE

MOD = DATA ARRAY USED TO DISPLAY FIELD TYPE LABELS

MSG = TEXT LABEL STORED AS PART OF DEFAULT FILE

NDEX = FILE TABLE USED BY MASS STORAGE ROUTINES, DIMENSIONED TO NUMBER OF TEXT MESSAGES + 1

NF = ACTUAL NUMBER OF REAL DATA ITEMS CONTAINED IN HACS DEFAULT FILE

NFLD = TOTAL NUMBER OF DATA FIELDS STORED IN HACS DEFAULT FILE (=NF+NI). DEFINITIONS OF EACH MESSAGE ARE STORED SEQUENTIALLY IN ARRAY LIST.

NI = ACTUAL NUMBER OF INTEGER DATA ITEMS CONTAINED IN HACS DEFAULT FILE

NW = LENGTH OF LAST RECORD, IN WORDS, READ FROM MESSAGE FILE

ONE = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 1

OUT = UNIT DEVICE NUMBER FOR OUTPUT FILE CREATED BY THIS PROGRAM

PACK = ARRAY CONTAINING PACKED MESSAGE TEXT DATA READ FROM MS FILE, MAXIMUM RECORD LENGTH OF 69. DIMENSIONED AS 70 WORD ARRAY TO ALLOW FORMAT CODE TO BE ADDED AFTER TEXT.

```

C      STRT  = SPECIAL FORMAT TAG FOR TYPE 3 MESSAGES, CREATES
C              DISPLAY IN UPDATE FORMAT WITH CODE 3 IN FIRST
C              POSITION, FIRST CHARACTER OF MESSAGE TEXT IN
C              SECOND POSITION
C      TERM  = DATA WORD CONTAINING CHARACTERS TO CLOSE MESSAGE
C              IN PACK FOR USE AS EXECUTION TIME FORMAT.
C      THR   = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 3
C      TWO   = DATA WORD CONTAINING TAG FOR CODED MESSAGES, TYPE 2
C      UDFLT = UNIT DEVICE NUMBER FOR EXTERNAL HACS DEFAULT FILE
C      UTXT  = UNIT DEVICE NUMBER FOR EXTERNAL MS MESSAGE TEXT FILE

C      AUTHOR - R.G. POTTS
C              ARTHUR D. LITTLE, INC.
C              35/318A ACORN PARK
C              CAMBRIDGE, MASS. 02140
C              TEL. 617-864-5770, EXT. 2813

C      DATE  - 20 AUGUST 1980

C      COMMON/BASE/MSG(10),MNF,MNI,NF,NI,LIST(275,6),
1      FVAL(225,3),IVAL(50,3)
C      REAL   MSG
C      DIMENSION DEFLT(2489)
C      EQUIVALENCE (DEFLT(1),MSG(1))

C      DIMENSION FPCK(71),MOD(2,2),NDEX(257),PACK(70)
C      INTEGER  FPCK,ONE,OUT,PACK,STRT,TERM,THR,TWO,UDFLT,UTXT
C      EQUIVALENCE (FPCK(2),PACK(1))
C      ODATA    FPCK(1)/10H(9X,      ',MOD(1,1)/4HINTE/,
1      MOD(1,2)/3HGER/,MOD(2,1)/4HREAL/,MOD(2,2)/1H /,
2      ONE/1H1/,OUT/12/,STRT/10H(5X,  '3/,TERM/2H"/,
3      THR/1H3/,TWO/1H2/,UDFLT/10/,UTXT/11/

C      -----READ EXTERNAL DEFAULT FILE CONTAINING FIELD NUMBERS AND
C      NAMES INDEXED SEQUENTIALLY. TOTAL NUMBER OF FIELDS ON FILE
C      IS SUM OF REAL AND INTEGER FIELDS.
C      REWIND UDFLT
C      READ(UDFLT) DEFLT
C      NFLD=NF+NI

C      -----WRITE INITIAL MESSAGE WITH STARTING MESSAGE NUMBER FOR
C      USE IN BATCH UPDATE
C      WRITE(OUT,1000)

C      -----OPEN MASS STORAGE MESSAGE FILE, INDEXED BY MESSAGE NUMBER,
C      AND START LOOP ON MESSAGES FROM 1 TO NFLD
C      CALL OPENMS(UTXT,NDEX,31,0)
C      DO 40 I=1,NFLD

C      -----DISPLAY MESSAGE NUMBER TOGETHER WITH HACS FIELD MODE, NUMBER
C      AND NAME.
C      IVAR=LIST(I,2)/1000
C      IVAR=IVAR+1
C      WRITE(OUT,1010) I,(MOD(IVAR,J),J=1,2),LIST(I,1),(LIST(I,J),J=3,5)

C      -----READ AND DECODE MESSAGE I
C      CALL READMS(UTXT,PACK,69,I)
C      NW=LENGTH(UTXT)
C      IF(PACK(1).EQ.ONE) GO TO 10
C      IF(PACK(1).EQ.TWO) GO TO 10
C      NW=NW+1
C      PACK(NW)=TERM
C      IF(PACK(1).EQ.THR) GO TO 20

C      -----OUTPUT UNCODED MESSAGE
C      WRITE(OUT,FPCK)
C      GO TO 30

C      -----WRITE CODED MESSAGES, TYPES 1 AND 2

```

```

10 WRITE(OUT,1020) PACK(1)
   GO TO 30
C-----
C-----WRITE CODED MESSAGE, TYPE 3, IN UPDATE FORMAT
20 PACK(1)=STR
   WRITE(OUT,PACK)
C-----
C-----WRITE MESSAGE DELIMITER, THEN CONTINUE LOOP FOR ALL MESSAGES.
30 WRITE(OUT,1030)
40 CONTINUE
   STOP
C-----
1000 FORMAT (42H000 = NUMBER OF LAST MESSAGE ON PRIOR FILE)
10100FORMAT (I3,21H = MESSAGE INDEX FOR ,A4,A3,7H FIELD ,I4,2X,3A4,
1 15H, MESSAGE TEXT:)
1020 FORMAT (5X,A1)
1030 FORMAT (5X)
   END
READY.

```

7.3.3 Model Text (File RGPEDT)

```
PROGRAM DISPLAY(OUTPUT,TAPE6=OUTPUT,TAPE12)
DIMENSION FPCK(192),NDEX(30),PACK(191)
INTEGER FPCK,PACK,TERM,UTXT
EQUIVALENCE (FPCK(2),PACK(1))
DATA FPCK(1)/10H(9X,
CALL OPENMS(UTXT,NDEX,30,0)
DO 40 I=1,29
CALL READMS(UTXT,PACK,190,I)
NW=LENGTH(UTXT)
NW=NW+1
PACK(NW)=TERM
WRITE(LP,1000) I
1000 FORMAT (//1X,"MESSAGE ",I2//)
WRITE(LP,FPCK)
40 CONTINUE
STOP
END
READY.
```

7.4 Utilities

Listings of two utility programs used for different purposes in the HACS/UIM development are given in Sections 7.4.1 and 7.4.2.

Section 7.4.1 contains descriptions and listings of 21 different sub-routines that together form a data compression utility package. The routines are used to pack (set) or unpack (read) numeric code values utilizing individual bits within memory words with a set of Fortran callable subroutines. Some of these routines have been incorporated within HACS/UIM. The complete package of code manipulation routines is very general, and should be readily adaptable to other applications requiring data compression.

Three levels of coding capability are provided:

- (1) Storing a string of single bit codes within a single memory word,
- (2) Storing a string of single bit codes within an array of memory words, and,
- (3) Storing a string of multiple bit codes within an array of memory words.

Within each level, the same functional procedures are provided for initialization, set, reset, test, pack and unpack operations. The set, reset and test features operate randomly on code N within a collection of coded values 1 to M. The pack and unpack features operate sequentially on all coded values, 1 to M, providing for bulk transfer of coded information. Detailed documentation of this utility package is given in Section 7.4.1.

In Section 7.4.2, a listing is given of a short utility program that was written to translate tab key entries (ASCII code) to corresponding control of stored file data (BCD code). Source files entered with an ASCII terminal contains lines of text in which the tab character may appear. In BCD mode, the ASCII tab is treated as any other character, and the source data line is unaffected. The version of the program given in Section 7.4.2 was used to process files containing Fortran source code with tab control entered to start comment line text in column 10, and source statement text in column 7.

The program reads each line of the input file, tests for the appearance of tab characters, and inserts spaces in the text as indicated. Both the input and output files are on disk; the output disk file can be listed after the program run to verify the desired tab spacing. Note that this program can readily be modified or adapted for processing similar data input formats to simplify keying operations.

7.4.1 Data Compression (File SAVCOD)

```
*****
*                                     *
*               DATA COMPRESSION UTILITY PACKAGE               *
*                                     *
*****
```

THE DATA COMPRESSION UTILITY PACKAGE IS COMPRISED OF A SERIES OF SUBROUTINES AND FUNCTIONS USED TO PACK OR UNPACK NUMERIC CODE VALUES UTILIZING INDIVIDUAL BITS WITHIN MEMORY WORDS. THREE LEVELS OF CAPABILITY ARE PROVIDED -

- (1) STORING A STRING OF SINGLE BIT CODES WITHIN A SINGLE MEMORY WORD,
- (2) STORING A STRING OF SINGLE BIT CODES WITHIN AN ARRAY OF MEMORY WORDS, AND,
- (3) STORING A STRING OF MULTIPLE BIT CODES WITHIN AN ARRAY OF MEMORY WORDS.

WITHIN EACH LEVEL, THE SAME FUNCTIONAL PROCEDURES ARE PROVIDED FOR INITIALIZATION, SET, RESET, TEST, PACK AND UNPACK CODES. THE SET, RESET AND TEST FEATURES OPERATE RANDOMLY ON CODE N WITHIN A COLLECTION OF CODED VALUES 1 TO M. THE PACK AND UNPACK FEATURES OPERATE SEQUENTIALLY ON ALL CODED VALUES, 1 TO M, PROVIDING FOR BULK TRANSFER OF CODED INFORMATION.

AUTHOR - R.G. POTTS, ARTHUR D. LITTLE, INC.,
35/318A ACORN PARK,
CAMBRIDGE, MASS., 02140
TEL. 617-864-5770 EXT. 2813

DATE - 17 NOVEMBER 1980

LEVEL (3) ROUTINES =

```
SUBROUTINE INIT(CODE,I,J,K)
LOGICAL FUNCTION ECHK(N)
SUBROUTINE SET(CODE,N,IVAL)
SUBROUTINE RSET(CODE,N)
FUNCTION ITST(CODE,N)
SUBROUTINE PACK(IAR,CODE)
SUBROUTINE UNPK(CODE,IAR)
```

LEVEL (2) ROUTINES =

```
SUBROUTINE INITH(CODE,I,J)
LOGICAL FUNCTION ECHKM(N)
SUBROUTINE SETH(CODE,N)
SUBROUTINE RSETH(CODE,N)
FUNCTION ITSTM(CODE,N)
SUBROUTINE PACKM(IAR,CODE)
SUBROUTINE UNPKM(CODE,IAR)
```

LEVEL (1) ROUTINES =

```
SUBROUTINE INIT1(CODE,I)
LOGICAL FUNCTION ECHK1(N)
SUBROUTINE SET1(CODE,N)
SUBROUTINE RSET1(CODE,N)
FUNCTION ITST1(CODE,N)
SUBROUTINE PACK1(IAR,CODE)
SUBROUTINE UNPK1(CODE,IAR)
```

FUNCTIONS ECHK, ECHKM AND ECHK1 ARE ONLY CALLED BY OTHER BIT MANIPULATION ROUTINES AND ARE NOT USED IN THE CALLING PROGRAM. REMAINING ROUTINES ARE USER CALLABLE AND ARGUMENT DEFINITIONS ARE GIVEN BELOW -

CODE = SINGLE WORD (LEVEL 1) OR INTEGER ARRAY (LEVELS 2 AND 3) FOR STORAGE OF PACKED CODED VALUES AT CPW CODED VALUES PER WORD. LEVELS 1 AND 2 USE SINGLE BIT CODED VALUES (0 OR 1). LEVEL 3 ROUTINES ASSUME MULTIPLE BIT CODED VALUES WITHIN A FIXED BYTE SIZE OF CDLN.
 I = INPUT TO INITIALIZATION ROUTINE SPECIFYING THE MAXIMUM NUMBER OF BITS IN EACH WORD OF CODE WHICH CAN BE USED FOR STORAGE OF CODED VALUES.
 IAR = INTEGER ARRAY CONTAINING UNPACKED CODED VALUES TO BE MOVED TO CODE (PACK OPERATION) OR TO BE RECEIVED FROM CODE (UNPACK OPERATION) IN SEQUENTIAL DATA TRANSFER.
 IVAL = NUMERIC EQUIVALENT OF CODE VALUE TO BE PACKED IN CODE POSITION N BY SUBROUTINE SET.
 J = MAXIMUM NUMBER OF WORDS IN ARRAY CODE WHICH ARE USED FOR STORAGE OF CODED VALUES. BY DEFINITION, THIS ARGUMENT APPLIES ONLY TO LEVEL 2 AND 3 ROUTINES AND A VALUE OF 1 IS USED FOR THE LEVEL 1 ROUTINES.
 K = DEFINES THE STORAGE REQUIRED FOR A SINGLE CODED VALUE FOR LEVEL 3 ROUTINES AT K BITS PER CODE, K BEING THE LARGEST BYTE REQUIRED TO CONTAIN THE LARGEST CODED VALUE. THIS DETERMINES THE ALLOWED INTEGER MAGNITUDE OF EACH CODED VALUE TO BE GREATER THAN OR EQUAL TO ZERO, AND LESS THAN $2^{*}K$. LEVEL 1 AND 2 ROUTINES USE SINGLE BIT CODING SO A VALUE OF 1 IS ASSUMED FOR K.
 N = CODED VALUE INDEX NUMBER, VARIES FROM 1 TO MAXIMUM NUMBER OF CODED VALUES IMPLIED BY I, J, K.

THE ARRAY IAR AND THE ARRAY CODE (EXCEPT FOR LEVEL 1) MUST BE DIMENSIONED IN THE CALLING PROGRAM ACCORDING TO THE LENGTHS IMPLIED BY I, J AND K.

THE INITIALIZATION ROUTINES (INIT, INITH AND INIT1) SPECIFY THE CHARACTERISTICS OF THE COMPRESSED DATA STORAGE. THE APPROPRIATE INITIALIZATION ROUTINE MUST BE CALLED ONCE BEFORE ANY REFERENCE TO A CODED VALUE BY ANY OF THE SET, RESET OR TEST ROUTINES, AND AGAIN ON RE-DEFINITION OF THE CODE WORD OR ARRAY OR ITS STORAGE CHARACTERISTICS. THE INITIALIZATION ROUTINES MUST ALSO BE CALLED IMMEDIATELY BEFORE EACH CALL TO ONE OF THE PACK OR UNPACK ROUTINES (REFER TO LISTINGS OF SUBROUTINES PACK AND UNPK FOR ADDITIONAL INFORMATION). THE INITIALIZATION PRE-SETS VALUES IN LABELLED COMMON WHICH ARE USED BY THE OTHER MANIPULATION ROUTINES. DEFINITIONS OF ALL VARIABLES IN COMMON USED BY THESE ROUTINES FOLLOW -

CDLN = INTEGER VALUE GIVING LENGTH OF SINGLE CODED VALUE IN BITS, DETERMINED FROM VALUE OF K (SEE ABOVE)
 CPW = MAXIMUM NUMBER OF CODED VALUES WHICH CAN BE STORED IN A SINGLE WORD (CODE)
 IERR = INTEGER ERROR FLAG RETURNED IN COMMON AS ZERO FOR NO ERROR, OR A VALUE IN THE RANGE 1 TO 9. IF MORE THAN ONE ERROR CAN OCCUR, ONLY THE LAST VALUE IS INDICATED. SPECIFIC ERROR CODES ARE DEFINED BELOW.
 ITMP = INTEGER WORD LOCATION USED AS SCRATCH SPACE FOR MANIPULATION ROUTINES.
 L = INDEX TO WORD OF ARRAY CODE CONTAINING CODED VALUE N (ASSUMED TO BE 1 FOR LEVEL 1 USE).
 MAXN = MAXIMUM NUMBER OF CODED VALUES AS DETERMINED DURING INITIALIZATION (REPLACED BY CPW IN LEVEL 1)
 MAXV = MAXIMUM VALUE OF A SINGLE CODE VALUE FOR LEVEL 3 (=1 FOR LEVELS 1 AND 2).
 SHFT = INTEGER FACTOR USED FOR POSITIONING SINGLE CODE VALUE RELATIVE TO PACKED CODE WORD.

COMMON VARIABLES ARE STORED IN THE LABELLED COMMON BLOCK NAMED GCODE, AND DIFFERENT LENGTHS ARE DEFINED FOR EACH GROUP OF MANIPULATION ROUTINES (ONLY REQUIRED VARIABLES ARE LISTED). HOWEVER ALL DEFINITIONS OF /GCODE/ ARE COMPATIBLE. FOR USER TESTING OF ERROR CONDITIONS, THE COMMON AREA MUST ALSO BE

DEFINED IN THE CALLING PROGRAM USING COMMON/GCODE/IERR WHERE IERR MAY BE ANY INTEGER VARIABLE. AFTER CALLING A MANIPULATION ROUTINE, A VALUE OF ZERO INDICATES NO ERROR, AND A NON-ZERO VALUE INDICATES ONE OR MORE ERRORS OCCURRED. LEVEL 1 ROUTINES RETURN ERROR CODES 1-6, LEVEL 2 RETURNS 1-7 AND LEVEL 3 RETURNS 1-9. DEFINITIONS OF ALL ERROR CODES FOLLOW -

- 0 = NO ERROR CONDITION DETECTED.
- 1 = LENGTH OF WORD IN BITS FOR CODE STORAGE IS ZERO OR NEGATIVE
- 2 = LENGTH OF WORD IN BITS REQUESTED FOR CODE STORAGE EXCEEDS MAXIMUM WORD LENGTH SET IN SUBROUTINES INIT, INITM AND INIT1 AS INSTALLATION PARAMETER.
- 3 = REQUESTED CODE POSITION N HAS A VALUE OF ZERO OR IS NEGATIVE.
- 4 = REQUESTED CODE POSITION N EXCEEDS THE NUMBER OF POSITIONS DEFINED FOR COMPACTED STORAGE.
- 5 = CODE VALUE REQUESTED TO BE STORED IS NEGATIVE (MUST BE ZERO OR POSITIVE). ZERO IS STORED.
- 6 = CODE VALUE REQUESTED TO BE STORED IS GREATER THAN MAXIMUM VALUE WHICH CAN BE STORED WITHIN LENGTH OF CODE SPECIFIED. MAXIMUM VALUE IS STORED.
- 7 = NUMBER OF WORDS REQUESTED FOR LENGTH OF ARRAY CODE IS NOT GREATER THAN ZERO.
- 8 = NUMBER OF BITS SPECIFIED FOR LENGTH OF SINGLE CODE VALUE IS NOT GREATER THAN ZERO.
- 9 = NUMBER OF BITS SPECIFIED FOR LENGTH OF SINGLE CODE VALUE EXCEEDS SPECIFIED LENGTH OF CODE WORD.

PROGRAM LISTINGS OF ALL SUBROUTINES AND FUNCTIONS FOLLOW. IN GENERAL, LEVEL 3 ROUTINES MAY BE USED TO PERFORM ALL OPERATIONS, HOWEVER FOR APPROPRIATE CODING STRUCTURES, LEVEL 2 AND 1 ROUTINES OFFER SIMPLIFIED EXECUTION FOR SIMPLIFIED CODING. ADDITIONAL NOTES ON USE ARE GIVEN IN INDIVIDUAL PROGRAM LISTINGS.

```
*****
*                                     *
*                                LEVEL 1 PROGRAM LISTINGS                                *
*                                     *
*****
```

SUBROUTINE INIT(CODE,I,J,K)

SUBROUTINE INIT INITIALIZES THE CODING ROUTINES TO STORE NEW CODES, OR TO READ PREVIOUSLY STORED CODES, IN THE INTEGER ARRAY CODE. THE ARRAY CODE, MUST BE DIMENSIONED IN THE CALLING PROGRAM TO BE OF LENGTH J OR GREATER. THE CHARACTERISTICS OF THE STORED NUMERIC CODES ARE SPECIFIED BY THE REMAINING ARGUMENTS -

- I = MAXIMUM NUMBER OF BITS IN EACH WORD OF THE ARRAY CODE WHICH CAN BE USED FOR STORAGE OF CODED VALUES.
- J = MAXIMUM NUMBER OF WORDS IN ARRAY CODE WHICH ARE USED FOR STORAGE OF CODED VALUES.
- K = DEFINES THE STORAGE REQUIRED FOR A SINGLE CODED VALUE TO BE FIXED LENGTH AT K BITS PER CODE. THIS DETERMINES THE ALLOWED INTEGER MAGNITUDE OF EACH CODED VALUE TO BE GREATER THAN OR EQUAL TO ZERO, AND LESS THAN 2**K.

ON RETURN, THE ERROR FLAG IERR IN COMMON IS ZERO IF NO ERRORS WERE ENCOUNTERED. ERROR CONDITIONS WILL CAUSE IERR TO BE SET TO 1,2,7,8 OR 9 ON RETURN, AND CONTROL VARIABLES IN COMMON TO BE SET FOR SINGLE BIT, SINGLE WORD CODE STORAGE.

SUBROUTINE INIT CONTAINS A SINGLE INTERNAL PARAMETER, MXWRD, WHICH DEFINES THE MAXIMUM ALLOWED UNSIGNED INTEGER WORD LENGTH IN BITS AND IS INSTALLATION DEPENDENT. FOR A NORMAL

```

C      16-BIT WORD LENGTH, MXWRD SHOULD BE SET TO 15.  FOR USE WITH
C      DOUBLE PRECISION (TWO-WORD) INTEGERS, MXWRD CAN BE SET TO 31
C      FOR A 16-BIT WORD LENGTH IF INTEGER SPECIFICATIONS ARE ALSO
C      MODIFIED IN THESE ROUTINES.  FOR USE ON THE CDC CYBERNET NET-
C      WORK, INTEGER ARITHMETIC IS LIMITED TO PARTIAL WORDS, SO MXWRD
C      IS SET TO 47 OUT OF 60 BITS AVAILABLE IN THE FULL WORD.

C      SUBROUTINE INIT MUST BE CALLED ONCE AND ONLY ONCE FOR EACH
C      CODED ARRAY PRIOR TO ALL CALLS USING THE ROUTINES SET, RSET
C      OR ITST WITH THE CODED ARRAY.  NOTE THAT INIT WILL CLEAR THE
C      CONTENTS OF THE REFERENCED CODED ARRAY.  INIT MUST BE CALLED
C      IMMEDIATELY BEFORE EACH CALL TO THE BULK TRANSFER ROUTINES
C      PACK AND UNPK.

C      COMMON VARIABLES USED - CDLN,CPW,IERR,L,MAXN,MAXV

C      SUBROUTINES REQUIRED - NONE

C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
C      INTEGER          CPW,CDLN,SHFT
C      INTEGER          CODE(1)
C      DATA            MXWRD/47/

C      -----TEST LENGTH OF WORD TO BE USED FOR CODE STORAGE.  CANNOT BE
C      LESS THAN 1 OR EXCEED MAXIMUM UNSIGNED INTEGER WORD LENGTH.
C      IF(I.LE.0) GO TO 20
C      IF(I.GT.MXWRD) GO TO 30

C      -----TEST NUMBER OF WORDS TO BE USED FOR CODE STORAGE.  CANNOT BE
C      LESS THAN 1.  UPPER LIMIT IS NOT TESTED SINCE THIS IS
C      CONTROLLED BY USER DIMENSION IN CALLING PROGRAM.
C      IF(J.LE.0) GO TO 40

C      -----TEST NUMBER OF BITS TO BE USED FOR SINGLE CODE.  CANNOT BE
C      LESS THAN 1 OR EXCEED SPECIFIED LENGTH OF CODE WORD.
C      IF(K.LE.0) GO TO 50
C      IF(K.GT.I) GO TO 60

C      -----NORMAL RETURN.  COMPUTE NUMBER OF CODES TO BE STORED PER
C      WORD (CPW), INITIALIZE ALL CODE WORDS TO ZERO, AND SET NORMAL
C      ERROR RETURN.  COMPUTE TOTAL NUMBER OF CODES WHICH CAN BE
C      STORED (MAXN), MOVE CODE LENGTH K TO COMMON VARIABLE CDLN,
C      AND COMPUTE MAXIMUM ALLOWED CODE VALUE (MAXV).
C      IERR=0
C      CPW=I/K
C      DO 10 L=1,J
C      10 CODE(L)=0
C      MAXN=CPW*J
C      CDLN=K
C      MAXV=2**CDLN-1
C      RETURN

C      -----ERROR RETURNS.  SET VALUE OF ERROR SWITCH IN COMMON AND
C      DEFAULT TO CODE DEFINITION USING SINGLE WORD CONTAINING CODES
C      ONE BIT IN LENGTH.
C      20 IERR=1
C      GO TO 70
C      30 IERR=2
C      GO TO 70
C      40 IERR=7
C      GO TO 70
C      50 IERR=8
C      GO TO 70
C      60 IERR=9
C      70 CPW=MXWRD
C      CODE(1)=0
C      MAXN=MXWRD
C      CDLN=1
C      MAXV=1

```

```

RETURN
END
LOGICAL FUNCTION ECHK(N)

```

```

LOGICAL FUNCTION ECHK (FOR ERROR CHECK) TESTS THE REQUESTED
CODE POSITION SPECIFIED BY THE ARGUMENT N. IF THE POSITION
IS NOT WITHIN THE ALLOWED NUMBER OF CODED VALUES (1 TO MAXN),
THE ERROR INDICATOR IERR IN COMMON IS SET TO 3 OR 4 AND THE
FUNCTION RETURNS A VALUE OF .TRUE. ALL OTHER VARIABLES IN
COMMON ARE UNCHANGED.

```

```

IF THE SPECIFIED CODE POSITION, N, IS VALID, THE ERROR CHECK
FUNCTION RETURNS A VALUE OF .FALSE. AND SETS VARIABLES IN
COMMON TO ACCESS THE VALUE OF THE NTH CODE PACKED IN AN ARRAY.
GIVEN N, THE LOCATION OF THE CODED VALUE IS DETERMINED BY THE
NUMBER OF CODED VALUES PER STORAGE WORD (CPW) AND THE LENGTH
OF EACH CODE (CDLN). BOTH CPW AND CDLN ARE DETERMINED ON
INITIALIZATION IN SUBROUTINE INIT. FOR ACCESSING THE REQUESTED
CODE THE FUNCTION RETURNS L AND SHFT. THE VALUE OF L IS THE
SUBSCRIPT INDEX TO THE WORD OF THE PACKED ARRAY CONTAINING
THE POSITION FOR THE CODED VALUE. SHFT IS AN INTEGER MULTI-
PLIER OR DIVISOR WHICH WILL MOVE A CODED VALUE OF LENGTH CDLN
TO OR FROM ITS POSITION IN WORD L FROM OR TO THE LOW ORDER
NUMERIC POSITION.

```

```

COMMON VARIABLES USED - CDLN,CPW,IERR,L,MAXN,SHFT,ITMP

```

```

SUBROUTINES REQUIRED - NONE

```

```

COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
INTEGER CPW,CDLN,SHFT
EQUIVALENCE (I,IPOS,ITMP)

```

```

C
C
C-----TEST REQUESTED CODE POSITION. MUST LIE WITHIN DEFINED
C      BOUNDARY OF CODE WORD STRUCTURE.
C      IF(N.LE.0) GO TO 10
C      IF(N.GT.MAXN) GO TO 20
C
C-----NORMAL RETURN. SET ERROR CODE AND FUNCTION VALUE.
C      IERR=0
C      ECHK=.FALSE.
C
C-----COMPUTE WORD ADDRESS (L) WITHIN CODE LIST ARRAY, AND POSITION
C      ADDRESS (IPOS) WITHIN WORD L FOR CODE LOCATION N.
C      I=N-1
C      L=I/CPW
C      IPOS=I-L*CPW
C      L=L+1
C
C-----COMPUTE SHIFT FACTOR TO ACCESS CODE N IN POSITION IPOS OF
C      WORD L.
C      I=CDLN*IPOS
C      SHFT=2**I
C      RETURN
C
C-----ERROR RETURNS.
C      10 IERR=3
C         GO TO 30
C      20 IERR=4
C      30 ECHK=.TRUE.
C         RETURN
C         END
C         SUBROUTINE RSET(CODE,N)
C
C      SUBROUTINE RSET (FOR RESET) IS USED TO CLEAR (I.E., SET TO
C      ZERO) CODE POSITION N FOR MULTIPLY VALUED CODES STORED IN AN
C      ARRAY. DEFINITION OF THE PACKED CODE STRUCTURE IS OBTAINED
C      FROM THE MOST RECENT CALL TO SUBROUTINE INIT. IF N IS INVALID,
C      IERR IS SET TO 3 OR 4 ON RETURN AND CODE IS UNCHANGED. OTHER-
C      WISE IERR=0 ON RETURN, AND ALL BITS WITHIN CODE N ARE RESET.
C
C
C
C

```

```

C
C
C      COMMON VARIABLES USED - ITMP,L,MAXV,SHFT
C      SUBROUTINES REQUIRED - ECHK
C
C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
C      INTEGER      CPW,SHFT,CDLN
C
C      INTEGER      CODE(1)
C      LOGICAL      ECHK
C
C-----CHECK ARGUMENT N, RETURN IF ERROR.
C      IF(ECHK(N)) RETURN
C
C-----MOVE MAXIMUM VALUE (ALL BITS SET) TO SHIFTED POSITION IN CODE.
C      ITMP=MAXV*SHFT
C
C-----RESET THIS POSITION IN WORD L OF CODE ARRAY.
C      CODE(L)=(.NOT.ITMP).AND.CODE(L)
C      RETURN
C      END
C      SUBROUTINE SET(CODE,N,IVAL)
C
C      SUBROUTINE SET STORES THE INTEGER VALUE IVAL AS A CODED
C      VALUE IN CODE POSITION N OF THE ARRAY CODE. DEFINITION OF
C      THE PACKED CODE STRUCTURE IS OBTAINED FROM THE MOST RECENT
C      CALL TO SUBROUTINE INIT. SUBROUTINE SET FIRST CLEARS THE
C      VALUE IN CODE POSITION N AND THEN STORES IVAL IF THE ARGUMENT
C      IS AN ACCEPTABLE CODE VALUE IN THE RANGE 1 TO MAXV. IF IVAL
C      IS LESS THAN ZERO, CODE N IS SET TO ZERO AND IERR=5 ON RETURN.
C      IF IVAL EXCEEDS MAXV, CODE N IS SET TO MAXV AND IERR=6 ON
C      RETURN. IERR MAY ALSO BE SET TO 3 OR 4 IF N IS INVALID AND
C      IN THIS CASE CODE IS UNCHANGED ON RETURN. IERR IS ZERO ON
C      NORMAL RETURN WITH NO ERRORS.
C
C      COMMON VARIABLES USED - IERR,ITMP,L,MAXV,SHFT
C      SUBROUTINES REQUIRED - RSET
C
C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
C      INTEGER      CPW,SHFT,CDLN
C
C      INTEGER      CODE(1)
C
C-----RESET CODE STORED IN POSITION N AND RETURN IF ERROR ENCOUNTERED
C      CALL RSET(CODE,N)
C      IF(IERR.NE.0) RETURN
C
C-----RETURN AFTER RESET IF IVAL IS LESS THAN OR EQUAL TO ZERO. IF
C      IVAL IS LESS THAN ZERO ALSO SET ERROR FLAG.
C      IF(IVAL.LT.0) IERR=5
C      IF(IVAL.LE.0) RETURN
C
C-----TEST FOR INPUT ARGUMENT LARGER THAN MAXIMUM ALLOWED CODED VALUE
C      IF(IVAL.GT.MAXV) GO TO 20
C
C-----MOVE CODED VALUE TO POSITION WITHIN WORD L OF CODED ARRAY,
C      STORE AND RETURN.
C      ITMP=IVAL*SHFT
C      10 CODE(L)=ITMP.OR.CODE(L)
C      RETURN
C
C-----IVAL EXCEEDS MAXIMUM ALLOWED CODED VALUE. SET ERROR FLAG THEN
C      STORE MAXIMUM VALUE.
C      20 IERR=6
C      ITMP=MAXV*SHFT
C      GO TO 10
C      END

```

FUNCTION ITST(CODE,N)

FUNCTION ITST RETURNS THE INTEGER VALUE OF CODE N STORED IN A PACKED ARRAY CODE. IF N IS NOT WITHIN THE RANGE OF THE PACKED CODES, A VALUE OF ZERO IS RETURNED FOR ITST AND IERR IS SET TO 3 OR 4. IF N IS VALID, THE VALUE OF ITST IS OBTAINED FROM THE PACKED CODE IN POSITION N IN THE RANGE 0 TO MAXV, AND IERR IS RETURNED AS ZERO. DEFINITION OF THE PACKED CODE STRUCTURE IS OBTAINED FROM THE MOST RECENT CALL TO SUBROUTINE INIT.

COMMON VARIABLES USED - ITMP,L,MAXV,SHFT

SUBROUTINES REQUIRED - ECHK

COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
INTEGER CPW,SHFT,CDLN

INTEGER CODE(1)
LOGICAL ECHK

-----INITIALIZE FUNCTION VALUE AND RETURN IF N IS INVALID.
ITST=0
IF(ECHK(N)) RETURN

-----MOVE CODED VALUE IN WORD L TO LOW ORDER POSITION OF ITMP.
ITMP=CODE(L)/SHFT

-----OBTAIN CODED VALUE BY REMOVING ANY BITS REMAINING IN HIGHER ORDER POSITIONS.
ITST=ITMP.AND.MAXV
RETURN
END
SUBROUTINE PACK(IAR,CODE)

SUBROUTINE PACK TRANSFERS DATA VALUES FROM THE INPUT ARRAY IAR TO COMPRESSED FORMAT IN THE ARRAY CODE. THE PARAMETERS OF THE COMPACTION MUST BE SET BY CALLING SUBROUTINE INIT WITH THE ARRAY CODE AS AN ARGUMENT PRIOR TO EACH NEW USE OF THIS ROUTINE. IF AN ERROR IS DETECTED BY SUBROUTINE INIT, THE CONTENTS OF THE OUTPUT ARRAY CODE ARE UNCHANGED.

IF NO ERRORS HAVE OCCURRED DURING INITIALIZATION, THE FIRST MAXN VALUES IN IAR ARE MOVED TO PACKED POSITIONS IN CODE. THE VALUES IN IAR ARE ASSUMED TO BE IN THE RANGE 0 TO MAXV. IF A NEGATIVE VALUE IS ENCOUNTERED, THE ERROR FLAG IERR IS SET TO 5, AND A CODE VALUE 0; ZERO IS STORED. IF A VALUE IN IAR IS GREATER THAN MAXV, THEN IERR IS SET TO 6 AND A CODE VALUE OF MAXV IS STORED. ON RETURN, IERR WILL BE ZERO (NO ERROR) OR SET TO THE LAST ERROR CONDITION (5 OR 6) ENCOUNTERED.

BASE = INTEGER QUANTITY USED AS A MULTIPLIER TO SHIFT
PACKED VALUE BY ONE CODE POSITION.
I = INDEX TO PACKED CODE POSITION WITHIN SINGLE WORD
OF ARRAY CODE, VARIES FROM 1 TO CPW.
IAR = INPUT INTEGER ARRAY. EACH WORD CONTAINS SINGLE
VALUE IN RANGE 0 TO MAXV TO BE PACKED INTO CODE.
ARRAY MUST BE DIMENSIONED IN CALLING PROGRAM TO
LENGTH OF MAXN OR GREATER.
J = INDEX TO ELEMENTS OF IAR TO BE PACKED, VARIES FROM
1 TO MAXN.

COMMON VARIABLES USED - CPW,IERR,ITMP,L,MAXN,MAXV,SHFT

SUBROUTINES REQUIRED - INIT (PRIOR CALL)

COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN,CDLN,MAXV
INTEGER CPW,CDLN,SHFT


```

C      (CPW), INITIALIZE ALL CODE WORDS TO ZERO, AND SET NORMAL ERROR
C      RETURN.  COMPUTE TOTAL NUMBER OF CODES WHICH CAN BE STORED
C      (MAXN).
      IERR=0
      CPW=1
      DO 10 L=1,J
10    CODE(L)=0
      MAXN=CPW*J
      RETURN

C-----ERROR RETURNS.  SET VALUE OF ERROR SWITCH IN COMMON AND
C      DEFAULT TO CODE DEFINITION USING SINGLE WORD CONTAINING CODES
C      ONE BIT IN LENGTH.
20    IERR=1
      GO TO 70
30    IERR=2
      GO TO 70
40    IERR=7
70    CPW=MXWRD
      CODE(1)=0
      MAXN=MXWRD
      RETURN
      END
      LOGICAL FUNCTION ECHKM(N)

CCCCCCCC
      LOGICAL FUNCTION ECHKM IS A SUB-SET OF LOGICAL FUNCTION ECHK.
      IN THIS VERSION, THE FUNCTION ASSUMES THE LENGTH OF CODED
      VALUES IS SET TO 1 FOR PACKING SINGLE BIT CODES INTO AN
      INTEGER ARRAY.  PROGRAM CODING IS SLIGHTLY SIMPLIFIED, BUT
      ALL OTHER FUNCTIONS AND USE REMAIN UNCHANGED.  REFER TO
      LISTING OF FUNCTION ECHK FOR ADDITIONAL INFORMATION.  NOTE
      THAT THE ECHK FUNCTIONS ARE ONLY USED INTERNALLY BY THE
      PACKING AND UNPACKING ROUTINES, AND CALLS TO ECHK, ECHKM
      AND ECHK1 ARE NOT INTERCHANGEABLE.  ON RETURN FROM ECHKM,
      THE ERROR FLAG IERR IS SET TO ZERO (NO ERRORS) OR TO 3 OR 4.

      COMMON VARIABLES USED - CPW,IERR,ITMP,L,MAXN,SHFT
      SUBROUTINES REQUIRED - NONE

      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
      INTEGER      CPW,SHFT
      EQUIVALENCE (I,IPOS,ITMP)

C-----TEST REQUESTED CODE POSITION.  MUST LIE WITHIN DEFINED
C      BOUNDARY OF CODE WORD STRUCTURE.
      IF(N.LE.0) GO TO 10
      IF(N.GT.MAXN) GO TO 20

C-----NORMAL RETURN.  SET ERROR CODE AND FUNCTION VALUE.
      IERR=0
      ECHKM=.FALSE.

C-----COMPUTE WORD ADDRESS (L) WITHIN CODE LIST ARRAY, AND POSITION
C      ADDRESS (IPOS) WITHIN WORD L FOR CODE LOCATION N.
      I=N-1
      L=I/CPW
      IPOS=I-L*CPW
      L=L+1

C-----COMPUTE SHIFT FACTOR TO ACCESS CODE N IN POSITION IPOS OF
C      WORD L.
      SHFT=2*IPOS
      RETURN

C-----ERROR RETURNS.
10    IERR=3
      GO TO 30
20    IERR=4
30    ECHKM=.TRUE.

```

```

RETURN
END
SUBROUTINE RSETM(CODE,N)

```

SUBROUTINE RSETH IS A SUB-SET OF SUBROUTINE RSET USED FOR SINGLE BIT CODES STORED WITHIN AN ARRAY, AND PROGRAM CODE IS SOMEWHAT SIMPLIFIED. BIT N IS RESET IF SET, UNCHANGED IF NOT SET. ON RETURN, IERR WILL BE SET TO 3 OR 4 IF N IS INVALID, OR ZERO IF N IS VALID.

COMMON VARIABLES USED - L, SHFT

SUBROUTINES REQUIRED - ECHKM

```
COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
INTEGER          CPW,SHFT
```

INTEGER	CODE(1)
LOGICAL	ECHKM

```

----CHECK ARGUMENT N, RETURN IF ERROR.
IF(ECHKM(N)) RETURN

```

```

-----USE COMPUTED WORD LOCATION AND SHIFT FACTOR TO RESET BIT N.
CODE(L)=(.NOT,SHFT).AND.CODE(L)
RETURN
END
SUBROUTINE SETM(CODE,N)

```

SUBROUTINE SETH IS A SUB-SET OF SUBROUTINE SET USED TO SET SINGLE BIT CODED VALUES STORED WITHIN AN ARRAY. SINCE ONLY A SINGLE VALUE CAN BE SET (I.E., 1), THE ARGUMENT IVAL IS ELIMINATED AND PROGRAM CODING IS GREATLY SIMPLIFIED. SUBROUTINE SETH WILL SET CODE POSITION N TO 1 IF UNSET, OR LEAVE IT UNCHANGED IF ALREADY SET. ON RETURN, IERR WILL BE SET TO 3 OR 4 IF N IS INVALID, AND CODE IS UNCHANGED ON RETURN. OTHERWISE, IERR=0 ON NORMAL RETURN.

COMMON VARIABLES USED - L,SHFT

SUBROUTINES REQUIRED - ECHKM

```
COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
INTEGER          CPW,SHFT
```

INTEGER	CODE(1)
LOGICAL	ECHKM

```

-----CHECK ARGUMENT N, RETURN IF ERROR.
IF(ECHKM(N)) RETURN

```

```

-----USE ARRAY WORD LOCATION L AND SHIFT TO CODE POSITION, SHFT,
      COMPUTED BY ECHKM TO SET BIT N.
CODE(L)=SHFT.OR.CODE(L)
RETURN
END
FUNCTION ITSTM(CODE,N)

```

FUNCTION ITSTM IS A SUB-SET OF FUNCTION ITST IN WHICH SINGLE BIT CODES ARE STORED IN AN ARRAY OF PACKED CODES. ON RETURN, ITSTM=1 IF BIT N IS SET, ZERO OTHERWISE. IF N IS NOT WITHIN THE RANGE OF PACKED CODES, A VALUE OF ZERO IS RETURNED FOR ITSTM AND IERR IS SET TO 3 OR 4. IF N IS VALID, IERR IS RETURNED AS ZERO.

COMMON VARIABLES USED - ITMP,L,SHFT

SUBROUTINES REQUIRED - ECHKM

```

C
C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
C      INTEGER      CPW,SHFT
C
C      INTEGER      CODE(1)
C      LOGICAL      ECHKM
C
C-----INITIALIZE FUNCTION VALUE AND RETURN IF N IS INVALID.
C      ITSTM=0
C      IF(ECHKM(N)) RETURN
C-----MOVE CODED VALUE IN WORD L TO LOW ORDER POSITION OF ITMP.
C      ITMP=CODE(L)/SHFT
C-----OBTAIN CODED VALUE BY STRIPPING OFF ANY HIGHER ORDER BITS
C      WHICH REMAIN.
C      ITSTM=ITMP.AND.1
C      RETURN
C      END
C      SUBROUTINE PACKM(IAR,CODE)
C
C      SUBROUTINE PACKM IS A SUB-SET OF SUBROUTINE PACK IN WHICH
C      VALUES STORED IN THE ARRAY IAR ARE PACKED INTO THE CODE WORD
C      ARRAY CODE. PACKM ASSUMES THAT THE VALUES STORED IN IAR ON
C      INPUT ARE ZERO OR ONE SO THAT ONLY SINGLE BIT VALUES ARE
C      PACKED INTO ARRAY CODE. THE PARAMETERS OF THE COMPACTION MUST
C      BE SET BY CALLING SUBROUTINE INITH WITH THE ARRAY CODE AS AN
C      ARGUMENT PRIOR TO EACH NEW USE OF THIS ROUTINE. ERROR CON-
C      DITIONS IF A VALUE IN IAR IS LESS THAN ZERO OR GREATER THAN
C      1 ARE THE SAME AS IN SUBROUTINE PACK.
C
C      I,IAR,J = REFER TO LISTING OF SUBROUTINE PACK FOR DEFINITIONS
C
C      COMMON VARIABLES USED - CPW,IERR,ITMP,L,MAXN,SHFT
C
C      SUBROUTINES REQUIRED - INITH (PRIOR CALL)
C
C      COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
C      INTEGER      CPW,SHFT
C
C      INTEGER      CODE(1)
C      DIMENSION    IAR(1)
C
C-----RETURN IF ERROR FLAG INDICATES PARAMETERS OF DATA COMPRESSION
C      NOT STORED IN COMMON, OTHERWISE INITIALIZE PACKING SEQUENCE.
C      IF(IERR.NE.0) RETURN
C      I=CPW
C      L=0
C-----OVERALL LOOP ON MAXN ELEMENTS OF IAR TO BE PACKED INTO CODE
C      DO 40 J=1,MAXN
C
C-----INCREMENT POSITION INDEX I. ADJUST SHIFT FACTOR TO NEXT
C      POSITION IF STILL IN SAME WORD. IF CURRENT WORD IS FULL, SET
C      TO PACK INTO FIRST POSITION OF NEXT WORD.
C      I=I+1
C      IF(I.LE.CPW) GO TO 10
C      L=L+1
C      SHFT=1
C      I=1
C      GO TO 20
C 10  SHFT=2*SHFT
C-----MOVE CURRENT VALUE IN IAR TO ITMP AND TEST MAGNITUDE. STORE
C      ONLY VALUES 0 OR 1. SET IERR TO 5 OR 6 IF IAR(J) IS NOT 0 OR 1
C      AND SKIP TO NEXT VALUE IF ITMP IS NOT GREATER THAN ZERO.
C 20  ITMP=IAR(J)
C      IF(ITMP.LT.0) IERR=5

```

```

IF(ITMP.LE.0) GO TO 40
IF(ITMP.GT.1) IERR=6
C-----STORE CODE VALUE IN PACKED POSITION AND CONTINUE TO END
C OF LOOP ON VALUES IN IAR INPUT ARRAY.
CODE(L)=CODE(L).OR.SHFT
40 CONTINUE
RETURN
END
SUBROUTINE UNPKM(CODE,IAR)
C
C SUBROUTINE UNPKM IS A SUB-SET OF SUBROUTINE UNPK AND MOVES
C SINGLE BIT CODED VALUES (0 OR 1) FROM A PACKED ARRAY CODE
C TO AN UNPACKED INTEGER ARRAY IAR. THE PARAMETERS OF THE
C COMPACTION MUST BE OBTAINED BY CALLING SUBROUTINE INITH WITH
C THE ARRAY IAR AS AN ARGUMENT PRIOR TO EACH NEW USE OF THIS
C ROUTINE. CODED VALUES PACKED IN THE ARRAY CODE ARE RETURNED
C AS MAXN INTEGER VALUES, EACH 0 OR 1, STORED IN WORDS 1 TO MAXN
C OF IAR. NO ERROR CONDITIONS ARE SET BY THIS ROUTINE.
C
C I,IAR,J = REFER TO LISTING OF SUBROUTINE UNPK FOR DEFINITIONS.
C
C COMMON VARIABLES USED - CPW,IERR,ITMP,L,MAXN
C
C SUBROUTINES REQUIRED - INITH (PRIOR CALL)
C
COMMON/GCODE/IERR,CPW,ITMP,SHFT,L,MAXN
INTEGER CPW,SHFT
C
INTEGER CODE(1)
DIMENSION IAR(1)
C
C-----RETURN IF ERROR FLAG INDICATES PARAMETERS OF DATA COMPRESSION
C NOT STORED IN COMMON, OTHERWISE INITIALIZE UNPACKING SEQUENCE.
IF(IERR.NE.0) RETURN
L=0
I=CPW
C
C-----OVERALL LOOP ON ELEMENTS OF IAR TO BE FILLED
DO 30 I=1,MAXN
C
C-----INCREMENT INDEX TO PACKED POSITION IN WORD OF ARRAY CODE. IF
C POSITIONED BEYOND CPW, SELECT FIRST POSITION OF NEXT WORD
C AND STORE ENTIRE CODE WORD IN ITMP. IF POSITIONED WITHIN
C CURRENT WORD, SHIFT CODE IN ITMP TO LOW-ORDER POSITION.
I=I+1
IF(I.LE.CPW) GO TO 10
L=L+1
ITMP=CODE(L)
I=1
GO TO 20
10 ITMP=ITMP/2
C
C-----MOVE CODED VALUE INTO IAR AND CONTINUE CYCLE UNTIL ALL ELEMENTS
C HAVE BEEN UNPACKED.
20 IAR(J)=ITMP.AND.1
30 CONTINUE
RETURN
END
C
C *****
C * LEVEL 3 PROGRAM LISTINGS *
C *****
C
SUBROUTINE INIT1(CODE,I)

```

```

C      SUBROUTINE INIT1 IS A SUB-SET OF SUBROUTINES INIT AND INITH.
C      IT IS EQUIVALENT TO AND REPLACEABLE BY INIT(CODE,I,1,1) OR
C      INITH(CODE,I,1). THAT IS, THIS ROUTINE FIXES THE LENGTH OF
C      PACKED CODED VALUES, K, TO 1 AS IN INITH, AND, IN ADDITION,
C      FIXES THE LENGTH OF THE ARRAY, J, TO 1 FOR STORAGE OF CODES
C      WITHIN A SINGLE WORD. PROGRAM CODING IS SIMPLIFIED, BUT ALL
C      OTHER FUNCTIONS AND USE REMAIN UNCHANGED. REFER TO LISTING OF
C      SUBROUTINE INIT FOR ADDITIONAL INFORMATION. ON RETURN, THE
C      ERROR FLAG IERR IS SET TO ZERO (NO ERRORS) OR TO 1 OR 2.

C      COMMON VARIABLES USED - CPW,IERR

C      SUBROUTINES REQUIRED - NONE

C      COMMON/GCODE/IERR,CPW,ITMP,SHFT
C      INTEGER      CPW,SHFT

C      INTEGER      CODE
C      DATA        MXWRD/47/

C      -----TEST LENGTH OF WORD TO BE USED FOR CODE STORAGE. CANNOT BE
C      LESS THAN 1 OR EXCEED MAXIMUM UNSIGNED INTEGER WORD LENGTH.
C      IF(I.LE.0) GO TO 20
C      IF(I.GT.MXWRD) GO TO 30

C      -----NORMAL RETURN. SAVE NUMBER OF CODES TO BE STORED IN SINGLE
C      CODE WORD (CPW), INITIALIZE CODE WORD TO ZERO AND SET NORMAL
C      ERROR RETURN.
C      IERR=0
C      CPW=I
C      10 CODE=0
C      RETURN

C      -----ERROR RETURNS. SET VALUE OF ERROR SWITCH IN COMMON AND
C      DEFAULT TO CODE DEFINITION FOR SINGLE WORD CONTAINING UP TO
C      MXWRD SINGLE BIT CODES.
C      20 IERR=1
C      GO TO 70
C      30 IERR=2
C      70 CPW=MXWRD
C      GO TO 10
C      END
C      LOGICAL FUNCTION ECHK1(N)

C      LOGICAL FUNCTION ECHK1 IS A SUB-SET OF FUNCTIONS ECHK AND
C      ECHKM. THAT IS, THE ROUTINE ASSUMES A FIXED LENGTH OF PACKED
C      CODED VALUES OF 1 AS IN ECHKM, AND, IN ADDITION, ASSUMES THAT
C      ALL PACKED CODED VALUES ARE STORED IN A SINGLE WORD, NOT IN AN
C      ARRAY. PROGRAM CODING IS SIMPLIFIED, BUT THE BASIC FUNCTION
C      AND USE OF THE ROUTINE ARE UNCHANGED. REFER TO LISTING OF
C      FUNCTION ECHK FOR ADDITIONAL INFORMATION. ON RETURN FROM
C      ECHK1, THE ERROR FLAG IERR IS SET TO ZERO (NO ERRORS) OR TO
C      3 OR 4.

C      COMMON VARIABLES USED - CPW,IERR,ITMP,SHFT

C      SUBROUTINES REQUIRED - NONE

C      COMMON/GCODE/IERR,CPW,ITMP,SHFT
C      INTEGER      CPW,SHFT
C      EQUIVALENCE (IPDS,ITMP)

C      -----TEST REQUESTED CODE POSITION. MUST LIE WITHIN DEFINED
C      BOUNDARY OF CODE WORD. NOTE CPW EXCHANGED FOR MAXN.
C      IF(N.LE.0) GO TO 10
C      IF(N.GT.CPW) GO TO 20

C      -----NORMAL RETURN. SET ERROR CODE AND FUNCTION VALUE.

```

```

IERR=0
ECHK1=.FALSE.
C-----COMPUTE SHIFT FACTOR TO CODE N POSITION WITHIN SINGLE PACKED
C CODE WORD.
IPOS=N-1
SHFT=2**IPOS
RETURN
C-----ERROR RETURNS
10 IERR=3
GO TO 30
20 IERR=4
30 ECHK1=.TRUE.
RETURN
END
SUBROUTINE RSET1(CODE,N)
C
C SUBROUTINE RSET1 IS A SUB-SET OF SUBROUTINES RSET AND RSETH
C USED FOR SINGLE BIT CODES STORED WITHIN A SINGLE CODE WORD.
C PROGRAM CODE IS SOMEWHAT SIMPLIFIED. BIT N IS RESET IF SET,
C UNCHANGED IF NOT SET. ON RETURN, IERR WILL BE SET TO 3 OR 4
C IF N IS INVALID, ZERO OTHERWISE.
C
C COMMON VARIABLES USED - SHFT
C SUBROUTINES REQUIRED - ECHK1
C
COMMON/GCODE/IERR,CPW,ITMP,SHFT
INTEGER CPW,SHFT
C
INTEGER CODE
LOGICAL ECHK1
C
C-----CHECK ARGUMENT N, RETURN IF ERROR.
IF(ECHK1(N)) RETURN
C
C-----USE COMPUTED SHIFT FACTOR TO RESET BIT N IN CODE WORD.
CODE=(.NOT.SHFT).AND.CODE
RETURN
END
SUBROUTINE SET1(CODE,N)
C
C SUBROUTINE SET1 IS A SUB-SET OF SUBROUTINES SET AND SETH. A
C SINGLE BIT CODED VALUE IS SET WITHIN A SINGLE CODE WORD. SINCE
C ONLY A SINGLE VALUE CAN BE SET (I.E., 1), THE ARGUMENT IVAL IS
C ELIMINATED AND PROGRAM CODING IS GREATLY SIMPLIFIED. SUB-
C ROUTINE SET1 WILL SET CODE POSITION N TO 1 IF UNSET, OR LEAVE
C IT UNCHANGED IF ALREADY SET. ON RETURN, IERR WILL BE SET TO
C 3 OR 4 IF N IS INVALID, AND CODE IS UNCHANGED ON RETURN.
C OTHERWISE IERR=0 ON NORMAL RETURN.
C
C COMMON VARIABLES USED - SHFT
C SUBROUTINES REQUIRED - ECHK1
C
COMMON/GCODE/IERR,CPW,ITMP,SHFT
INTEGER CPW,SHFT
C
INTEGER CODE
LOGICAL ECHK1
C
C-----CHECK ARGUMENT N, RETURN IF ERROR.
IF(ECHK1(N)) RETURN
C
C-----USE SHIFT TO CODE POSITION N COMPUTED BY ECHK1 TO SET BIT N.
CODE=SHFT.OR.CODE
RETURN

```

END
FUNCTION ITST1(CODE,N)

FUNCTION ITST1 IS A SUB-SET OF FUNCTIONS ITST AND ITSTM IN *
WHICH SINGLE BIT CODES ARE STORED IN A SINGLE PACKED CODE WORD.
ON RETURN, ITST1=1 IF BIT N IS SET, ZERO OTHERWISE. IF N IS
NOT WITHIN THE RANGE OF PACKED CODES, A VALUE OF ZERO IS
RETURNED FOR ITST1 AND IERR IS SET TO 3 OR 4. IF N IS VALID,
IERR IS RETURNED AS ZERO.

COMMON VARIABLES USED - ITMP,SHFT

SUBROUTINES REQUIRED - ECHK1

COMMON/GCODE/IERR,CPW,ITMP,SHFT
INTEGER CPW,SHFT

INTEGER CODE
LOGICAL ECHK1

-----INITIALIZE FUNCTION VALUE AND RETURN IF N IS INVALID.
ITST1=0
IF(ECHK1(N)) RETURN

-----MOVE CODED VALUE IN CODE WORD TO LOW ORDER POSITION OF ITMP.
ITMP=CODE/SHFT

-----OBTAIN CODED VALUE BY STRIPPING OFF ANY HIGHER ORDER BITS
WHICH REMAIN.
ITST1=ITMP.AND.1
RETURN
END
SUBROUTINE PACK1(IAR,CODE)

SUBROUTINE PACK1 IS A SUB-SET OF SUBROUTINES PACK AND PACKM.
THAT IS, INPUT VALUES OF ZERO OR ONE STORED IN THE ARRAY IAR
ARE PACKED INTO A SINGLE WORD, CODE. THE PARAMETERS OF THE
COMPACTION MUST BE SET BY CALLING SUBROUTINE INIT1 WITH CODE
AS AN ARGUMENT PRIOR TO EACH NEW USE OF THIS ROUTINE. ERROR
CONDITIONS IF A VALUE IN IAR IS LESS THAN ZERO OR GREATER THAN
1 ARE THE SAME AS IN SUBROUTINE PACK.

IAR,J = REFER TO LISTING OF SUBROUTINE PACK FOR DEFINITIONS

COMMON VARIABLES USED - CPW,IERR,ITMP,SHFT

SUBROUTINES REQUIRED - INIT1 (PRIOR CALL)

COMMON/GCODE/IERR,CPW,ITMP,SHFT
INTEGER CPW,SHFT

INTEGER CODE
DIMENSION IAR(1)

-----RETURN IF ERROR FLAG INDICATES PARAMETERS OF DATA COMPRESSION
NOT STORED IN COMMON, OTHERWISE INITIALIZE PACKING SEQUENCE.
IF(IERR.NE.0) RETURN
J=1
SHFT=1

-----RETURN HERE FOR EACH NEXT VALUE IN IAR TO BE PACKED. MOVE
VALUE FROM IAR TO ITMP AND TEST MAGNITUDE. STORE ONLY VALUES
0 OR 1. SET IERR TO 5 OR 6 IF IAR(J) IS NOT 0 OR 1 AND THEN
SKIP TO NEXT VALUE IF ITMP IS NOT GREATER THAN ZERO.

10 ITMP=IAR(J)
IF(ITMP.LT.0) IERR=5
IF(ITMP.LE.0) GO TO 30
IF(ITMP.GT.1) IERR=6

```

C-----STORE CODE VALUE IN PACKED POSITION.
CODE=CODE.OR.SHFT
C-----INCREMENT INDEX TO INPUT WORDS IN IAR. RETURN AFTER CPW WORDS
C      HAVE BEEN PACKED, OTHERWISE INCREMENT SHIFT FACTOR AND CONTINUE
C      CYCLE TO PACK NEXT VALUE.
30 J=J+1
   IF(J.GT.CPW) RETURN
   SHFT=2*SHFT
   GO TO 10
END
SUBROUTINE UNPK1(CODE,IAR)

SUBROUTINE UNPK1 IS A SUB-SET OF SUBROUTINES UNPK AND UNPKM
AND MOVES SINGLE BIT CODED VALUES (0 OR 1) FROM A SINGLE
PACKED WORD (CODE) TO AN UNPACKED INTEGER ARRAY IAR. THE
PARAMETERS OF THE COMPACTION MUST BE OBTAINED BY CALLING
SUBROUTINE INIT1 WITH THE ARRAY IAR AS AN ARGUMENT PRIOR TO
EACH NEW USE OF THIS ROUTINE. CODED VALUES PACKED IN THE
SINGLE WORD CODE ARE RETURNED AS CPW INTEGER VALUES, EACH 0 OR
1, STORED IN WORDS 1 TO CPW OF IAR. NO ERROR CONDITIONS ARE
SET BY THIS ROUTINE.

IAR,J = REFER TO LISTING OF SUBROUTINE UNPK FOR DEFINITIONS.
COMMON VARIABLES USED - CPW,IERR,ITMP
SUBROUTINES REQUIRED - INIT1 (PRIOR CALL)

COMMON/GCODE/IERR,CPW,ITMP,SHFT
INTEGER      CPW,SHFT
C
INTEGER      CODE
DIMENSION    IAR(1)
C
C-----RETURN IF ERROR FLAG INDICATES PARAMETERS OF DATA COMPRESSION
C      NOT STORED IN COMMON, OTHERWISE MOVE CODE WORD TO TEMPORARY
C      LOCATION FOR MANIPULATION.
IF(IERR.NE.0) RETURN
ITMP=CODE
C-----LOOP ON ELEMENTS OF IAR TO BE FILLED. SET EACH TO LOW-ORDER
C      VALUE OF ITMP, THEN SHIFT ITMP ONE POSITION. CONTINUE LOOP
C      UNTIL ALL CPW ELEMENTS OF IAR ARE SET.
DO 10 J=1,CPW
  IAR(J)=ITMP.AND.1
10 ITMP=ITMP/2
  RETURN
END

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

```

READY.

7.4.2 Tab Key (File SAVCNV)

```

PROGRAM CNV(TAPE10,TAPE11)
INTEGER BLK,BUFF(80),COM,OUTR(80),TAB
DATA BLK/1H /,COM/1HC/,TAB/1H~/
REWIND 11
10 READ(10,1000) BUFF
1000 FORMAT (80A1)
IF(EOF(10)) 90,15
15 CONTINUE
IF(BUFF(1).NE.COM) GO TO 50
OUTB(1)=COM
IF(BUFF(2).NE.TAB) GO TO 40
DO 20 I=2,9
20 OUTB(I)=BLK
M=10
N=4
IF(BUFF(4).NE.TAB) GO TO 80
DO 30 I=10,20
30 OUTB(I)=BLK
M=21
N=6
GO TO 80
40 M=2
N=2
GO TO 80
50 IF(BUFF(1).NE.TAB) GO TO 70
DO 60 I=1,6
60 OUTB(I)=BLK
M=7
N=3
GO TO 80
70 M=1
N=1
80 OUTB(M)=BUFF(N)
M=M+1
N=N+1
IF(M.LE.80) GO TO 80
WRITE(11,1010) OUTB
1010 FORMAT (80A1)
GO TO 10
90 ENDFILE 11
REWIND 11
END
READY.

```