A STUDY PROGRAM ON THE DEVELOPMENT OF MATHEMATICAL
MODEL(S) FOR MICROBIAL BURDEN PREDICTION

JPL Contract 952532

Volume VII

User's Manual for the

Input Translator Program

(Task 2 of Phase VIII)

Prepared by:

Lloyd B. Farabee
Program Manager

for

Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91103

Approved:

George E. Fosdick
Chief
Operations Analysis

C. Curlander
Manager
Systems Engineering

Martin Marietta Corporation
P. O. Box 179
Denver, Colorado 80201
As a result of additional work done in Phase VIII (JPL Contract 952532), three new volumes have been prepared. Volume VII describes the Input Translator Program (ITP), which prepares input data for the MBPM. Volume VIII contains changes to Volume VI and includes a complete listing of all subroutines in the MBPM, as revised in April 1970. Volume IX describes the use of the ITP with the MBPM in predicting the microbial burden on the Mariner Mars 69-3.

(Sheet to be pasted in Volume II, page ii)

This work was performed for the Jet Propulsion Laboratory, California Institute of Technology, sponsored by the National Aeronautics and Space Administration under Contract NAS7-100.
FOREWORD

This document describes the work performed as Task 2 of Phase VIII of JPL Contract 952532. (This is a follow-on to JPL Contract 952028.) Phase VIII accomplished improvements in the preparation of input data for the computer programs generated in Phase III.

This document is a supplement to the Revised User's Manual for the Microbial Burden Prediction Model (Volume VI). As the work described in this report is a continuation of the work described in references 1 through 6, this document is written on the assumption that the reader is familiar with the material in the referenced reports.

This work was performed for the Jet Propulsion Laboratory, California Institute of Technology, sponsored by the National Aeronautics and Space Administration under Contract NAS7-100.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreword</td>
<td>1</td>
</tr>
<tr>
<td>Contents</td>
<td>ii</td>
</tr>
<tr>
<td>Illustrations</td>
<td>iii</td>
</tr>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. Technical Discussion</td>
<td>2</td>
</tr>
<tr>
<td>A. General</td>
<td>2</td>
</tr>
<tr>
<td>B. Description of Subroutines</td>
<td>3</td>
</tr>
<tr>
<td>C. Input Data</td>
<td>9</td>
</tr>
<tr>
<td>D. Output Data</td>
<td>16</td>
</tr>
<tr>
<td>E. Control Cards</td>
<td>16</td>
</tr>
<tr>
<td>III. Results and Conclusions</td>
<td>20</td>
</tr>
<tr>
<td>References</td>
<td>22</td>
</tr>
<tr>
<td>Appendix</td>
<td>A1</td>
</tr>
</tbody>
</table>
ILLUSTRATIONS

Figure

1. Input Translator Program Macrologic .................. 4

Table

1. Data Cards .................................................. 6
2. Initial Run Through ITP, TAS, and BPS ............... 17
3. Restart Run Through ITP, TAS, and BPS ............. 18
4. Initial TAS Run Without Rerunning ITP .............. 19
5. Restart TAS Without Rerunning ITP ................. 19
I. INTRODUCTION

During Phases I, II, and III of JPL Contract 952028 the requirements for the Microbial Burden Prediction Model (MBPM) were established, and the model was developed and programmed for computer utilization. During Phases IV, V, and VI, the model was used to estimate the burden on the Mariner V Spacecraft. References 1 through 5 describe this work.

As a result of the above work, a number of changes were identified that would improve the effectiveness of the existing computer programs. These changes were effected during Phase VII of the carry-on contract (JPL Contract 952532). As a result of the experience gained during the preparation of the input data, the desirability of a more manageable method of preparing this data for input to the program was recognized.

The Input Translator Program (ITP) has been developed during Phase VIII of JPL Contract 952532 to meet this latter need. The ITP performs much of the repetitious and time-consuming work involved in preparing input data. In addition, it also performs bookkeeping chores such as maintaining the current status of all parts. The ITP has been programmed so that only minor revisions have been made to the previously developed portions of the program. In most cases, an error of input data to the ITP is printed as an error, but does not stop the production of an input tape for the Tape Alteration Subroutine (TAS). The manual preparation of input data is therefore reduced to coding the JPL Quality Assurance Daily Activity Report information for key punching, key punching this information, and making corrections to the errors which are flagged by the ITP.

This volume is a User's Manual which describes the various subroutines comprising the Input Translator Program, the method of data input, and the process of transferring the output data to the MBPM.
The principal factor in the development of the ITP that contributed to changes in existing programs was the change from making parts inputs at the Task level to making these inputs at the Subtask level.

Although the changes are minor in nature and small in number, the effect is felt on a number of pages of the User's Manual. The changes to these pages are listed in Volume VIII of the series of reports.

The ITP has been used to prepare input data based on the Mariner Mars 69-3 spacecraft for use on the MBPM. Using this input data, the MBPM has been run on the CDC 6400 at Martin Marietta's Denver facility and on the Univac 1108 at JPL in Pasadena.

Volume IX of this series of reports describes the preparation of the input data, the results of the burden prediction of the MM 69-3, and a comparison with swab sample data taken during the spacecraft preparation.

II. TECHNICAL DISCUSSION
A. General

Development of the Input Translator Program was constrained by two considerations: (1) the input format would be on the four types of cards as developed by JPL to express all the operations covered by the QA log during the assembly, test and checkout of a Mariner type of spacecraft, and (2) the output format would require a minimum amount of changes to be made to the Microbial Burden Prediction Model and the subroutines already developed as a part of the model.

A STAGE as presently used is a segment of the preparation, usually several days, devoted to a type of activity such as systems test, space simulation test, etc.. The STAGE is listed by the ITP and the number is carried as a matter of record, but is not a significant item to the ITP.
A TASK is now, by definition, equal to one day. The TASK number (of the STAGE) is carried on the record, and the Day number (of the entire Run) is merely reproduced from the input data and is not significant to the ITP. Time is input to the program in minutes, and the ITP checks the total time of all SUBTASKS within the TASK. If the total does not equal 1440 minutes (24 hours) an error message is printed.

The SUBTASK has become the basic unit of the program, rather than the OPERATION. The revisions to the MBPM are principally due to this change in approach, and are described in the Addendum to the User's Manual for the Microbial Burden Prediction Model (Ref. 7) issued concurrently with this report, but under separate cover. Omission of pertinent information (environment code, operation number, time, part number, etc.) will be printed as an error, but will not cause program termination. An error will be printed if the sum of the OPERATION times within the SUBTASK is not equal to the input SUBTASK time.

The only unique parameters still associated with the OPERATION are the type and time. All other information is common to the entire SUBTASK.

Figure 1 shows a flow diagram of the ITP macrologic. Input data are normally on cards, while output data are on tape. The quantity of output data prepared for a spacecraft program requires the use of tape. For example, the ITP output data for the Mariner Mars 69-3 program amounts to almost 60,000 records. The output tape generated by the ITP is compatible with the input requirements of the TAS subroutine.

B. Description of Subroutines

A description of each of the subroutines in the Input Translator Program is given below. A complete listing of each of these subroutines is included in the Appendix.
FIGURE 1.
INPUT TRANSLATOR PROGRAM
MACROLOGIC
**SUBROUTINE ITP**, the Input Translator Program (subroutine). This program prepares a Tape 12 for input into the TAS subroutine (Ref. Vol. VI, pp. 18-19). The ITP, with a minimal amount of input, performs the tedious preparation of input required by the TAS and subsequently by the BPS (Ref. Vol. VI, p. 21). The ITP, by performing much of the routine work, including the bookkeeping and cross-checking to detect errors and inconsistencies, prepares input data that are more free of errors than may be expected from manually prepared input data. Input data to the ITP are much less repetitive than required by the TAS, consisting of only four types of cards which must be input in a specified order. The types of cards are (1) stage cards, (2) task cards, (3) subtask cards, and (4) parts cards. From these cards the ITP prepares all necessary input data except for environment, basic distributions, and operations data. Instruction types and data are described in the Input Data Section. The card types are identified in Table 1 and fully described in Reference 6, pp. 26-38.

**SUBROUTINE CREAD**, the Card READING subroutine. This subroutine reads one card under the appropriate FORMAT statement and handles writing of control cards.

**SUBROUTINE DDDQ**, the DD and DQ card building subroutine. This subroutine handles the buildup of all DD and DQ data cards, or data records on Tape 12.

**SUBROUTINE ERROR**, the ERROR writing subroutine. This subroutine writes error messages which call attention to items which should be checked for input accuracy. As the errors normally encountered in the ITP are non-fatal, a tape is prepared which is acceptable to the TAS and BPS.
### TABLE 1

Data Cards

<table>
<thead>
<tr>
<th>CC</th>
<th>CONTROL CARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>RD</td>
<td>RUN DESCRIPTION CARD</td>
</tr>
<tr>
<td>SD</td>
<td>STAGE DESCRIPTION CARD</td>
</tr>
<tr>
<td>TD</td>
<td>TASK DESCRIPTION CARD</td>
</tr>
<tr>
<td>EM</td>
<td>ENVIRONMENTAL SURFACE LIFETIME MODIFIERS</td>
</tr>
<tr>
<td>ED</td>
<td>ENVIRONMENTAL DESCRIPTION CARD</td>
</tr>
<tr>
<td>EQ</td>
<td>ENVIRONMENTAL QUANTITIES CARD</td>
</tr>
<tr>
<td>OD</td>
<td>OPERATION DESCRIPTION CARD</td>
</tr>
<tr>
<td>OQ</td>
<td>OPERATION QUANTITIES CARD</td>
</tr>
<tr>
<td>KD</td>
<td>SUBTASK DESCRIPTION CARD</td>
</tr>
<tr>
<td>PD</td>
<td>PART DESCRIPTION CARD</td>
</tr>
<tr>
<td>PQ</td>
<td>PART QUANTITIES CARD</td>
</tr>
<tr>
<td>DD</td>
<td>DISTRIBUTION DESCRIPTION CARD</td>
</tr>
<tr>
<td>DQ</td>
<td>DISTRIBUTION QUANTITIES CARD(S)</td>
</tr>
<tr>
<td>KC</td>
<td>SUBTASK CHANGE CARD</td>
</tr>
<tr>
<td>KØ</td>
<td>SUBTASK OPERATION CARD</td>
</tr>
<tr>
<td>PE</td>
<td>PART EFFECT CARD</td>
</tr>
<tr>
<td>ZD</td>
<td>ZONE DESCRIPTION CARD</td>
</tr>
<tr>
<td>ZC</td>
<td>ZONE COMPOSITION CARD</td>
</tr>
</tbody>
</table>
SUBROUTINE FINAL, the FINAL analysis subroutine. This subroutine performs a final analysis when an END card is encountered in CREAD. This analysis includes a list of parts which currently are not installed and a summary of the initial input area and the calculated area for each subzone.

SUBROUTINE HEAD, HEADING subroutine. This subroutine prints a heading on each page consisting of parameter numbers along the top of the page for ease in identifying parameters to be changed by TAS.

FUNCTION IOPPER, is a function to determine if an operation is a decontamination type of operation.

FUNCTION ISUBZO, is a function to obtain a number from the subzone letter code.

SUBROUTINE KC, the KC card building subroutine. This subroutine builds all KC cards except for the KC cards written to transfer areas from one surface to another surface of the same zone (referred to as "O" and "E" cards).

SUBROUTINE KOPe, the KO and PE card building subroutine. This subroutine builds all KO cards and their respective PE cards.

SUBROUTINE NVIR, is the subroutine that obtains environment numbers from letter codes.

SUBROUTINE OEKC, is the subroutine that builds a complete subtask card set (principally KC cards) for switching subzone occluded areas from exposed to occluded (Ø code) or from occluded to exposed (E code).
**SUBROUTINE PDINTL**, is the PD and PQ card building subroutine for subzones. This subroutine is called on an initial run to build the PD and PQ cards for the zones (parts 1 to 100).

**SUBROUTINE PDPQ**, is the PD and PQ card building subroutine for parts (which are sometimes entire subzones). This subroutine builds PD and PQ cards for parts or subzones which are to be installed or removed.

**SUBROUTINE PNPUT**, is the Preliminary INPUT subroutine. This subroutine is called on an initial run to read in the following:

1. the parts list
2. the surface histograms
3. the distribution shape for part contact retention
4. the surface retention factor for fallout contamination
5. the mean part contact retention
6. the tool retention distribution for contact
7. the mean number of organisms on hand or tool
8. the retention factor for the tool in determining contamination by contact
9. the percent of total area for contact on exposed surfaces
10. the distribution shape of the fraction of organisms removed
11. the mean fraction of organisms removed from all surfaces
(12) up to ten decontamination operations

(13) environment codes

SUBROUTINE RECWS, is the RECord Writing Subroutine. This subroutine writes the data records on Tape 12. It also prints the data records for inspection and ease in identifying the records to be altered by TAS.

C. Input Data

The input data cards consist of an INITIAL card, a set of cards describing conditions which are needed throughout the program, and a set of cards describing the operations, in sequence, which comprise the entire assembly and testing preceding launch.

The INITIAL card has INITIAL in Columns 1-7 for an initial run and RESTART in Columns 1-7 for a restart run. (The program reads only Columns 1-6.) On restart runs a Tape 2 (Ref. Vol. VI, p. 18) must also be used.

The next set of cards are the PARTS LIST INPUT, the SURFACE HISTROGRAM INPUT, the CONSTANT LIST, the DECONTAMINATE OPERATIONS CARD, and the ENVIRONMENT CONVERSION CARD.

PARTS LIST INPUT

FORMAT (1X, I2, A1, I2, 3X, 4A6, 1X, 4F9.0, 4X, A6)

The first set of card inputs consists of the parts (and zones) list. Each part (and zone) is identified by a two-digit number (zone), one letter (subzone), and a two-digit number (part); for example, 01A01. The ITP limits use to twenty zone numbers (1 to 20), five subzone codes (A, B, C, D, and E), and sixty parts (1 to 60) per subzone, in addition to the subzone codes (e.g.,
16A00). A Burden Prediction Subroutine (BPS) zone (the subzone is the largest item recognized by the program and is often referred to in this report as a zone) is identified by the first three characters, followed by two zeroes, for example, 01B00, which corresponds to part 21, in the memory matrix and in the printout, 20D00, which corresponds to part 80, and 09C00, which corresponds to part 49. For each part and zone (a zone is treated as a part) a card must be read in which contains the part code (15A00, 02C25, 09E60, etc.), a 24-alphameric character description, an area for each of the four surfaces (in square inches, right justified in field), and an optional 6-alphameric character serial number. A zone card must be followed immediately by parts cards for all the parts in the zone. A blank card terminates the list.

SURFACE HISTOGRAM INPUT

FORMAT (I10) (First card of each Set)

This number (right justified) is the number of intervals in the histogram.

FORMAT (8E10.3) (Second card of each set)

This consists of one or two DR cards and one or two XR cards. For description of these input data see Vol. VI, pp. 34-35.

Two histograms must be input, one for each of the exposed surface types. The "top exposed" must be input first (even though no top exposed surface areas exist) and the "exposed" is input second.
CONSTANT LIST

FORMAT (I10, 2E10.3, I10, 4E10.3, /, I10, 4E10.3)

Several required constants are read in as follows: (Ref. Vol. VI, pp. 33-34, 37-38)

IAB - corresponds to IAB(6) on PQ (part quantities) card
AAG - corresponds to AAG on PQ card
AAS - corresponds to AAS on PQ card
LS - corresponds to LS on PE (part effect) card
APC - corresponds to APC on PE card
APS - corresponds to APS on PE card
APA(L) - a percentage to be multiplied times the area of surface 1 to obtain contact area on PE card
LK - corresponds to LK on PE card
AR(J) - corresponds to AR(J) on PE card

DECONTAMINATE OPERATIONS CARD

FORMAT (I1I5)

Operation numbers which are decontamination operations are listed as

I1, IDET(1), IDET(2), . . . . . , IDET(I1)

where I1 is the number of decontaminate operations in the subtask. This number cannot exceed 10.

IDET(J) is the operation number associated with the decontamination.

ENVIRONMENT CONVERSION CARD

FORMAT (10,(1X, A1))

Up to ten environment letter codes ordered so as to correspond to the environment numbers to be used in place of the code.
INVIR(1), . . . . . , INVIR(10)

The next set of cards are the STAGE CARD, the TASK CARD, the SUBTASK CARD, and the PARTS CARD and must appear in that order the first time. Subsequently, any one of the first three cards may follow the PARTS CARD and the order of all following cards must be maintained each time. Maintenance of this order is very important, as any deviation will cause a termination of the run.

STAGE CARD

Columns 1-5
STAGE

Columns 8-9
The stage number. A right-justified integer, unique for this run.

Columns 11-52
42 alphameric character stage description.

TASK CARD

Columns 1-4
TASK

Columns 5-7
The task number. A right-justified integer, unique for this stage.

Columns 9-50
42 alphameric character task description.

SUBTASK CARD

Columns 1-2
The subtask number. A right-justified positive integer, not necessarily unique.

Columns 3-26
24 alphameric character subtask description.

Columns 27-32
Right-justified number of total minutes to perform subtask.

Columns 33-34
Number of men performing subtask. A right-justified integer. If a zero is input for installation of removal subtask, the program sets the number equal to one.

Column 35
Alphameric character to indicate environment code. If blank, or code is not in the input list, the environment is assumed to be unchanged from the preceding subtask.
Column 50  Alphameric character to signify type of subtask:  
I = Install. All parts (not zones) marked "X" will be installed.

R  Remove. All parts (not zones) marked "X" will be removed.

M  Modify. The purpose of this subtask is to change zone areas from surface 2 to 4 or vice versa.

All other characters have no meaning to the program.

Columns 51-52  First operation number in subtask. A right-justified integer.

Columns 53-56  Time (in minutes) required for operation. A right-justified number.

Columns 57-58  Second operation number.

Columns 59-62  Time for second operation.

Columns 63-64  Third operation number.

Columns 65-69  Time for third operation.

Columns 69-70  Fourth operation number.

Columns 71-74  Time for fourth operation.

Columns 75-76  Fifth operation number.

Columns 77-80  Time for fifth operation.

When an operation number of zero is encountered, the program assumes that there are no more operations for that subtask.

PARTS CARDS

Each card has thirteen 6-alphameric fields (1-6), (7-12), ......, (73-78). Each field has the five-character part number in the first five columns. The first two columns identify the zone, the next column identifies the subzone, and the last two columns identify the part number.

Column 6 of each field should contain one letter, as follows:
If column 6 is blank the zone is considered to have secondary or adjacent effect (fallout but no contact on the PE cards).

If column 6 is "X" or "W", the zone is considered to have primary effect (fallout and contact on the PE cards). If the type of subtask is "I" or "R" the X'ed part or W'ed zone is installed or removed.

If column 6 is "∅" the occluded area of the zone is moved from the exposed (surface 2) where it has been accumulating burden, to the occluded portion (surface 4) of the zone. An "∅" code must be encountered for each zone sometime during the run. All subsequent parts installed in this zone will have the occluded area added to the occluded portion of the zone rather than to the exposed.

If Column 6 is "E" the occluded area of the zone, which at some previous time had been moved to occluded with an "∅" code, is moved back to the exposed area of the zone. On subsequent part installations the occluded portion of the part will be added to the exposed portion of the zone. If an "E" code is used on a zone, the last code encountered before the program terminates should be an "∅" code. An example of an "E" code would be the exposure of an area when a cover is removed. This area had previously been switched to occluded when the cover was installed (with an "∅" code).
Other Input Considerations

A number of points need to be called to the attention of the user of the program. These are:

(1) A task consists of one day or 1440 minutes.

(2) The program will accept up to 1000 subtasks per task. As the largest number used is 99, these will not necessarily be unique numbers. Only positive numbers are used.

(3) The mated area for a part is included in the occluded area for the part unless the occluded area is shown as zero.

(4) All X'ed parts on an "I" or "R" subtask are installed or removed. Zones may not be installed or removed with an "X" (in the sixth column of the six-column field on the PARTS CARD). Any part with a "W" in the sixth column is considered to be a zone and the total zone is installed or removed. Contact part effect is assumed for "W" code as for "X" code.

(5) Part effect cards are written only for zones. The primary effect applies to zones containing parts X'ed or W'ed. A cumulative area is kept for each surface of each zone so that the proper contact area is credited.

(6) When a part is installed, the occluded area of the part is added to the exposed (surface 2) area of the zone unless a (ZONE CHANGE CODE) (Ø CODE) is encountered for that zone.

(7) If a part is installed (or removed) two or more consecutive times an error is printed but the part is assumed to be already installed (or removed).
(8) An operation with zero time is printed out as an error even though this may be a legitimate input.

(9) A tape able to execute in TAS and BPS is normally made regardless of the errors encountered. The errors are printed as information on input data that should be checked for revision.

(10) If an environment code is omitted, the environment code for the preceding subtask is used.

(11) If no men are specified for an installation or removal subtask, the program assumes one man was involved.

D. Output Data

A list of the status (whether installed or not) of all parts and the surface areas of all zones is maintained constantly and may be printed out after any task by exercising the option. Parts may be installed or removed in any subtask; entire subzones may be installed or removed in any subtask; the occluded area of any subzone may become exposed, or, if exposed, may become occluded in any subtask; environments may be changed at any subtask; the program has restart capability after any task; the ITP output data is put on tape in the format acceptable to the subroutines of the MBPM as a "Tape 12". (Ref. Vol. VI, pp. 19-21.) A description of each of these record types is given in Vol. VI, pp. 26-38, as amended by Reference 7.

E. Control Cards

As originally programmed, TAS uses a Tape 12 for input. This is still the case, and TAS can be operated from a Tape 12 whether generated as before or generated by the ITP. A Tape 12 must therefore be written by ITP whenever output is desired. Table 2 shows the control cards for an initial run through ITP, TAS, and BPS. The output from ITP is saved on Tape 12 so that additional initial runs may be made in the future without rerunning ITP.
### ITP INITIAL DATA

<table>
<thead>
<tr>
<th>OIADD</th>
<th>SUBZONE STRUCTURE (A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OIA01</td>
<td>OCTAGON STRUCTURE ASSEMBLY</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.0</td>
<td>0.8</td>
<td>5</td>
<td>200.0</td>
</tr>
<tr>
<td>15</td>
<td>0.0</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ITP STANDARD DATA

**STAGE 1 SPACECRAFT ASSEMBLY AND CHECKOUT**

**TASK 1 DAY 001 6/17/68**

<table>
<thead>
<tr>
<th>IINST UPPER RING HARN</th>
<th>510 281 1 2 511</th>
<th>I 1 400 2 4014 70</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 SECURED IN HIGH BAY</td>
<td>795 H2</td>
<td>3 15T18 795</td>
</tr>
<tr>
<td>D00000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TAS DATA

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 MARINER HARS 1969-3(MM 69-3) ASSEMBLY/TEST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>87</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

**FIN.**

---

**Table 2.** INITIAL RUN THROUGH ITP, TAS AND BPS
RUN LBBITP,06G24S,07G95,40,4G3100. ITP L BUSCH CALL X4999

MSG MOUNT TAPE XXXXX TO READ LBB (PROGRAM TAPE)

ASG,TAPE,Y,XXXXX

ASG,TPF /// 500

COPING TAPE

FREE TAPE

MSG MOUNT TAPE XXXXX TO READ LBB (RESTART DUMP READ)

ASG,T 2,T,XXXXX

MSG MOUNT TAPE XXXXX TO WRITE LBB (RESTART DUMP WRITE)

ASG,T 3,T,XXXXX

MSG MOUNT TAPE XXXXX TO WRITE LBB (TO SAVE OUTPUT)

ASG,T 12,T,XXXXX

FOR MAIN

CALL ITP
CALL TAS
CALL BPS
RETURN
END

ITP STANDARD DATA

RESTART

TASK 4 DAY 004 6/20/68
1ROUTE HARN 70 3H2 1 2 512 3 15I 1 25 2 2514 2D
01A02 15A01 15A02 15A06X

13SECURE IN TENT 830 T2 8 12T17 830
00000

TAS DATA

99999 0 2

FIN.

TABLE 3. RESTART RUN THROUGH ITP, TAS & BPS
TABLE 4. INITIAL TAS RUN WITHOUT RERUNNING ITP

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>1 0 1 0 0 1 6 1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>2 1 2</td>
<td>69</td>
</tr>
<tr>
<td>2</td>
<td>1 8</td>
<td>0 2</td>
<td>87</td>
</tr>
<tr>
<td>99999</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 5. RESTART TAS WITHOUT RERUNNING ITP
If the ITP is to be stopped for restart later a Tape 3 must be written. Table 2 shows the cards for writing the restart dump. Then, when restarting ITP the previously written Tape 3 is read as a Tape 2 as shown on Table 3. Table 3 shows the control cards for a restart run through ITP, TAS, and BPS. The restart dump from the previous run is read from Tape 2 and a restart dump is written on Tape 3 for subsequent runs. The ITP output is being saved on Tape 12 so that it will not be necessary to rerun ITP on future restarts. The future restarts will have to be restarted at this same place, however. As the Univac computer at JPL is not presently configured for multitape operation on FORTRAN, restart must be used whenever the output of ITP exceeds approximately 26,000 records.

No changes can be made in ITP initial data on restart. ITP standard data for tasks subsequent to the restart are not affected by the restarting. Changes in TAS data (for example, changing the environmental conditions) are made by inserting the new TAS data (identified by record number from the preceding portion of the run) immediately following the END card of the ITP data. The format of this new data card is described on page 12 of Volume VI. Even when no changes are made, a TAS data card must be added (for example, see "99999" card on Table 5).

Table 4 shows the cards for an initial run on TAS using ITP output data saved on a Tape 12, with restart dump written on Tape 3. Table 5 shows a restart on TAS using ITP output data saved on a Tape 12, reading restart dump from Tape 2 and writing another restart dump on Tape 3.

III. RESULTS AND CONCLUSIONS

The Input Translator Program meets all of the specified objectives. Checkout procedures, trial runs and a complete run of the MM 69-3 have been made with the ITP. Input errors are, in general, printed out for information,
but do not cause termination of the program.

The program has been run satisfactorily on the CDC 6400/6500 at Martin Marietta's Denver facility and on the Univac 1108 at JPL. When the program was used on the 1108, however, it was discovered that multitape operation is not presently possible with FORTRAN programs (March 1970). With the amount of data generated on the MM 69-3, at least two (possibly three) tapes would be required to process the entire run on a single pass through the computer. This means that the run must be subdivided into sections which the 1108 can handle by exercising the RESTART option of the ITP.

The ITP has been used to make a complete run of the MM 69-3, from preparation of input data from the QA log to a final burden prediction. The results of this run are reported in Reference 8.
REFERENCES

The following references, which were issued in December 1968 and are available through the NASA STAR, form the final report for JPL Contract 952028, A Study Program on the Development of a Mathematical Model(s) for Microbial Burden Prediction:

3. Volume III, Appendices
5. Volume V, Addendum: Appendices

The following reference, which was issued in October 1969 forms the report for the work performed under the original phase of JPL Contract 952532 prior to the work covered by this report:


The following two references, which are issued concurrently with this document, when combined with this document comprise the report for the work performed under the extension to JPL Contract 952532.

APPENDIX

PROGRAM SOURCE DECK

LISTINGS
SUBROUTINE ITP

PROGRAM ITP ( INPUT TRANSLATOR PROGRAM ) PREPARES INPUTS
FOR THE MBPH ( MICROBIAL BURDEN PREDICTION MODEL ) PROGRAM

COMMON AAG * AAS * AMEN * APA(2) *
* APC * APS * AR(4) * AREA(4,13) *
* BLANK * BUMP(4,11,2) * FAREA(20,5,4) * IAB *
* ICODE(13) * IDET(10) * IERROR * IEX *
* IFLG * IJPRT * INVIR(10) * INSTAL *
* IOP(5) * IPART(11,1001) * IPOINT(20,5,61) * IPRT(13) *
* IREM0V * IRR(13) * ISTAR * ISZONE(13) *
* ITYPE * IX(6) * IW * IZERO *
* IZONE(13) * I1 * JPRT * KS *
* KT * 1K * LR * LS *
* M * 1N * NBUMP(4) * NP *
* NPARTS * NVIRON * N2 * STIME *
* TIME(5) * XX(22) * ZERO

DIMENSION APart(11,1001)
EQUIVALENCE ( IPART(1,1) , APART(1,1) )
EQUIVALENCE ( XX(21) , TOTAL ) , ( XX(22) , ATIME )
DIMENSION DUM(17816)
EQUIVALENCE ( AAG , DUM(1) )
DATA IM / 1HM /
DATA KOUNT / 0 /

REWRITE 12
READ (5,10) INP
10 FORMAT ( A6 )
IF ( INP NE. GHRESTAR ) GO TO 25

KOUNT = 1
REWRITE 2
READ (2) NAME ( DUM(J) , J = 1 , 17816 )
IFL = 6
IF ( NAME NE. GHITP ) GO TO 50
WRITE (6,15)
15 FORMAT ( 4Z1****ERROR**** NO ITP RESTART DUMP FOUND )
STOP

25 CALL PNPUT
JPRT = 3
50 CALL CREAD
GO TO ( 75 , 100 , 200 , 300 , 100 ) * IFLG
75 CALL FINAL
RETURN
100 CALL RECWS
GO TO 50
200 IF ( ITYPE NE. IM ) GO TO 50
IX(1) = N
IX(3) = IZERO
IX(5) = IZERO
IX(6) = IZERO
GO TO ( 201 , 202 , 203 ) * JPRT
201 IX(2) = 1
IX(4) = 2
JPRT = 2
GO TO 205
202 IX(2) = 2
IX(4) = 1
JPRT = 1
GO TO 205
203 IX(2) = 0
   IX(4) = 1
   JPRE = 1
205 DO 210 I = 5, 7
   XX(I) = BLANK
210 CONTINUE
   LR = 2
   CALL RECWS
   TOTAL = ZERO
   DO 215 I = 1, 5
   TOTAL = TOTAL + TIME(I)
215 CONTINUE
   IF ( TOTAL .GT. ATIME + .5 ) OR. TOTAL .LT. ATIME - .5 )
      CALL ERROR (15)
   GO TO 50
300 IF ( ITYPE .EQ. IM )
   IF ( KOUNT .EQ. IZERO )
      CALL PDPO
      CALL DDPQ
      CALL KC
      CALL KOPE
500 CALL DEKC
      GO TO 50
END
SUBROUTINE CREAD

CARD READING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2),
* APC, APS, AR(4), AREA(4,13),
* BLANK, BUMP(4,11,2), FAREA(20,5,4), IAB
* ICODE(13), IDET(10), IERROR, IEEX
* IF6, IJPR, INVIR(10), INSTAL
* IOP(5), IPART(11,1001), IPOINT(20,5,61), IPRT(13)
* IREMOV, IRR(13), ISTART, ISZONE(13)
* ITYPE, IX(6), IW, IZERO
* IZONE(13), I1, JPRE, KS
* KI, KL, LR, LS
* M, N, NBUMP(4), NP
* NPARTS, NVIRON, N2, STIME
* TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUVALENCE (IPART(1,1), APART(1,1))
EQUVALENCE (XX(22), ATIME)
DIMENSION DUM(17816)
EQUVALENCE (AAG, DUM(1))
DIMENSION IWORD(14)
DATA IBLANK /4H/,
* ITASK /4HTASK/,
* IEND /4HEND/,
* ISTOP /4HSTOP/,
* END /5HENDOF/,
* TASK /6HTASK/,
* DATA NAME /5HNAME/,

GO TO (10, 200, 300, 10, 10, 10) IF6
10 READ (5,20) (IWORD(I), I = 1, 14)
10 READ (5,20) (IWORD(I), I = 1, 8)
C 20 FORMAT (13AE, A2)
20 FORMAT (8AE0)
C DECODE (IWORD(1), 30) ICOL
DECODE (4, 35, IWORD(1)) ICOL
30 FORMAT (A4)
IF (ICOL .EQ. ITASK) GO TO 50
IF (ICOL .EQ. IBLANK) GO TO 10
IF (ICOL .EQ. ISTOP) GO TO 100
IF (ICOL .NE. IEND) GO TO 40

IF6 = 1
DO 35 I = 1, 5
IX(I) = IZERO
XX(I+1) = BLANK
35 CONTINUE
C XX(1) = END
XX(2) = TASK
LR = 2
CALL RECWS
DO 37 I = 1, 7
XX(I) = ZERO
37 CONTINUE
C LR = 1
CALL RECWS
C NP = 70
REWIND 3
WRITE (3) NAME, (DUM(J), J = 10, 17816)
END FILE 12
END FILE 12
REWIND 12
RETURN

C 40 DECODE (IWORD(I), 265) N, (XX(I), I = 1, 4) ATIME
C AMEN, NVIRON, IYPE, (IOP(I), TIME(I), I = 1, 5) UNIVAC
C
40 DECODE (80, 265, IWORD(1)) N, (XX(I), I = 1, 4) ATIME
C AMEN, NVIRON, IYPE, (IOP(I), TIME(I), I = 1, 5) CDC
C
STIME = ATIME
IFLG = 3
RETURN

50 IF (IFLG .EQ. 5) GO TO 70
DO 60 J = 1, 6
IX(J) = IZERO
XX(J) = BLANK
60 CONTINUE
XX(7) = BLANK
IF (KT .EQ. IZERO .OR. IFLG .EQ. 6) GO TO 65
LR = 2
XX(1) = END
XX(2) = TASK
CALL RECWS

55 LR = 1
IF (IFLG .NE. 6) GO TO 67
IX(1) = IFLG
CALL RECWS

67 IX(1) = 3
IX(2) = 3
IF (KT .EQ. IZERO) IX(2) = 1

66 CALL RECWS

70 KSAVE = KT

C 80 FORMAT (A4, I3, IX, 7AE) UNIVAC
C
80 FORMAT (A4, I3, IX, 7AE) CDC

C 80 DECODE (IWORD(I), 80) MN, KT, (XX(I), I = 1, 7) UNIVAC
C DECODE (50, 80, IWORD(1)) MN, KT, (XX(I), I = 1, 7) CDC

85 CONTINUE
IF (KSAVE .EQ. IZERO) GO TO 90
IF (STIME .LT. 1439.5 .OR. STIME .LT. 1440.5) CALL ERROR (9)

90 STIME = ZERO
IFLG = 2
RETURN

100 DO 110 J = 1, 6
IX(J) = IZERO
XX(J) = BLANK
110 CONTINUE
IF (KT .EQ. IZERO .OR. IFLG .EQ. 6) GO TO 115
LP = 2
XX(1) = END
XX(2) = GO TASK
CALL RECWS

115 LR = 1
IF (IFLG .NE. 6) GO TO 117
IX(1) = IFLG
CALL RECWS

117 IX(1) = 2
IX(2) = 3
IF ( IFLG .NE. 1 )  GO TO 120
  IX(1) = 1
  IX(2) = 1
  IX(4) = 1
120 CALL RECWS
  IF ( IFLG .NE. 1 )  GO TO 140
  LR = 6
  IX(1) = 1
  IX(2) = IZERO
  XX(1) = RHRUN DE
  XX(2) = EHSUPRINT
  XX(3) = RHION CA
  XX(4) = RHRD
  DO 130 J = 5, 7
    XX(J) = BLANK
130 CONTINUE
  CALL RECWS
C 140 DECODE ( IWORD(1), 15G )
140 FORMAT ( A6, I3, 1X, 7AG )
  MN, KS, ( XX(I), I = 1, 7 )
  UNIVAC
150 DECODE ( 52, 15G, IWORD(1) )
150 FORMAT ( A6, I3, 1X, 7AG )
  MN, KS, ( XX(I), I = 1, 7 )
  CDC
  LR = 7
  IX(1) = KS
  DO 160 I = 2, 6
    IX(I) = IZERO
160 CONTINUE
  IFLG = 5
  RETURN
200 READ (5,205) N, ( XX(I), I = 1, 4 ), ATIME, AMEN, NVIRON
  , ITPF, ( IGF(I), TIME(I), I = 1, 5 )
205 FORMAT ( I2, 4AG, F5.0, F2.0, A1, 14X, A1, 5 ( I2, F4.0 ) )
  STIME = STIME + ATIME
  IFLG = 3
  RETURN
300 READ (5,305) ( IZONE(I), ISZONE(I), IPRT(I), ICODE(I), I = 1, 13 )
305 FORMAT ( 26 ( I2, A1 ) )
  IJPR = 13
  DO 31C I = 1, 13
    IF ( IZONE(I) .NE. IZERO )  GO TO 310
    IF ( ISZONE(I) .NE. 1HC .AND. ISZONE(I) .NE. IH )  GO TO 310
    IF ( IPRT(I) .EQ. IZERO )  GO TO 315
310 CONTINUE
  GO TO 320
315 IJPR = I - 1
320 IFLG = 4
  RETURN
END
SUBROUTINE DDDD

DD AND DQ CARD BUILDING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2), APC, APS, AR, AREA(I:13), BLANK, BUMP(I:10), FAREA(I:5), IAB, ICODE(I:13), IDET(I:10), IERROR, IEX, IFLG, IJPR, INVIR(I:10), INSTAL, IOP(I:5), IPART(I:1001), IPOINT(I:20), IPR(I:13), IREMOV, IRR(I:13), ISTART, ISZONE(I:13), IZONE(I:13), I1, JPRE, KS, KT, LK, LR, LS, M, N, NBRMP(I:4), NP, NPARTS, NVIRON, N2, STIME, TIME(I:5), XX(I:22), ZERO

DIMENSION APART(I:1001), DESC(I:2), DUM(I:5)

EQUIVALENCE ( IPART(I:1), APART(I:1) )

EQUIVALENCE ( DESC(I:1), DUM(I:1) )

DATA ( DUM(I:1), I = 1, 5 ) / 6H TOP, 6H EXT S, 6H EXTE, 6HROR S, 6HRFACE /

IF ( IJPR(I) .LE. ZERO ) GO TO 600
DO 500 I = 1, IJPR(I)
IF ( IRR(I) .NE. ZERO ) GO TO 500
I1 = IZONE(I)
I2 = ISUBZ0 ( ISZONE(I) )
JSTART = IPOINT(I1-I2)
IJ = 4
IF ( APART(I3-JSTART) .LT. ZERO ) IJ = 3
DO 10 K = 1, 6
IX(K) = ZERO
10 CONTINUE
IF ( ICODE(I) .NE. IW ) GO TO 20
JSTART = JSTART
OCCL = AMAX1 ( ( AREA(I:1) - AREA(I:3) ) .LE. ZERO )
GO TO 25
20 IJ3 = IPRT(I) + 1
JSTART = IPOINT(I1-I2-I3)
OCCL = AMAX1 ( ( APART(I3-JSTART) - APART(I-JSTART) ) .LE. ZERO )
GO TO 400
25 IJ = 1
IF ( ICODE(I) .NE. IW ) GO TO 35
ASURF = APEA(I)
GO TO 50
35 ASURF = APART(I-J+I3)
50 IF ( J .EQ. 2 .AND. IJ .EQ. 3 ) ASURF = ASURF + OCCL
IF ( J .EQ. 4 ) ASURF = OCCL
IF ( ASURF .EQ. ZERO ) GO TO 400
IF ( IJ1 .EQ. IREMOV ) ASURF = ZERO
I = 16 + 4 * I + J
KJ = MIND(I,2)
SUBROUTINE ITP

PROGRAM ITP (INPUT TRANSLATOR PROGRAM) PREPARES INPUTS FOR THE MBBM (MICROBIAL BURDEN PREDICTION MODEL) PROGRAM

COMMON AAG, AAS, AMEN, APA(2),
       APC, APS, AR(4), AREA(40,13),
       BLANK, BUMP(4,11,2), FAREA(20,4), IAB
       ICODE(13), IDET(10), IERROR, IEX
       IFLG, IPRT, INVR(10), INSTAL
       IOP(5), IPART(11,1001), IPOINT(20,5,61), IPRT(13)
       IREMREV, IR(13), ISTART, ISZONE(13),
       ITYPE, IX(6), IW, IZERO
       IZONE(13), II, JPRE, KS
       LT, LK, LR, LS
       M, N, &BUHP(4), NP
       NPARTS, NVIRON, N2, STIME
       TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUIVALENCE (IPART(1), APART(1)),
EQUIVALENCE (XX(2), TOTAL), (XX(22), ATIME)
DIMENSION DUM(17816)
EQUIVALENCE (AAG, DUM(1))
DATA IMC / 1HM /
DATA KOUNT / 0 /

C
REWIN 12
READ(5,10) INP
10 FORMAT(AG)
   IF (INP .NE. 6HRESTAR) GO TO 25

C
   KOUNT = 1
   REWIN 2
   READ(2) NAME( DUM(J), J = 1, 17816)
   IFLG = 6
   IF (NAME .EQ. 6HITP) GO TO 50
   WRITE(6,15)
   15 FORMAT(42H1********ERROR******** NO ITP RESTART DUMP FOUND)
   STOP

C
25 CALL PNPUP
   JPRE = 3
5G CALL CREAD
   GO TO (75, 100, 200, 300, 100, 100) IFLG
75 CALL FINAL
   RETURN
10G CALL REWS
   GO TO 50
200 IF (ITYPE .EQ. IM) GO TO 50
   IX(1) = N
   IX(3) = IZERO
   IX(5) = IZERO
   IX(6) = IZERO
   GO TO (201, 202, 203) JPRE
201 IX(2) = 1
   IX(4) = 2
   JPRE = 2
   GO TO 205
202 IX(2) = 2
   IX(4) = 1
   JPRE = 1
   GO TO 205
203 IX(2) = 0
IX(4) = 1
JPRE = 1
205 DO 210 I = 5, 7
XX(I) = BLANK
210 CONTINUE
LR = 2
CALL RECS
TOTAL = ZERO
DO 215 I = 1, 5
TOTAL = TOTAL + TIME(I)
215 CONTINUE
IF ( TOTAL .GT. ATIME + .5 .OR. TOTAL .LT. ATIME - .5 )
   CALL ERROR (15)
   GO TO 50
300 IF ( ITYPE .EQ. IM )
   IF ( KOUNT .EQ. IZERO )
      CALL PDPG
      CALL DODO
      CALL KC
      CALL KOPE
      GO TO 50
      END
SUBROUTINE CREAD

CARD READING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2)
AFC, APS, AR(4), AREA(4,13)
BLANK, BUMP(4,11,2), FAREA(20,5,4), IAB
ICODE(13), IDET(10), IERROR, IEX
IFLG, JPRT, INVIR(10), INSTAL
IOP(5), IPART(1), IPOINT(20,5,61), IPRT(13)
IREMOVE, IRR(13), ISTART, ISZONE(13)
ITYPE, IX(6), IW, IZERO
IZONE(13), I1, JPRE, KS
K, LK, LR, LS
M, N, NBUMP(4), NP
NPARTS, NVIRON, N2, STIME
TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUIVALENCE (IPART(1,1), APART(1,1))
EQUIVALENCE (XX(22), ATIME)
DIMENSION DUM(17816)
EQUIVALENCE (AAG, DUM(11))
DIMENSION IWORD(14)
DATA IBLANK /4H/ *
ITASK /4H/ *
IEND /4H/ *
ISTAG /4H/ *
END /4H/ *
TASK /6H/ *
DATA NAME /6H/ *

GO TO ( 10, 200, 300, 10, 10, 10 ), IFLG
10 READ (5,20) (IWORD(I), I = 1, 14)
20 FORMAT (13A6, A2)
20 FORMAT (4A10)
C DECODE (IWORD(I), 30) ICOL
DECODE (4, 3D, IWORD(1)) ICOL
3D FORMAT ( A4 )
IF (ICOL.EQ.ITASK) GO TO 50
IF (ICOL.EQ.IBLANK) GO TO 10
IF (ICOL.EQ.ISTAG) GO TO 100
IF (ICOL.NE.IEND) GO TO 40

IFLG = 1
DO 35 I = 1, 6
IX(I) = IZERO
XX(I+1) = BLANK
35 CONTINUE

XX(I) = END
XX(2) = TASK
LR = 2
CALL RECWS
DO 37 I = 1, 7
XX(I) = ZERO
37 CONTINUE

LR = 1
CALL RECWS

NP = 70
REWRITE (3) NAME, ( DUM(J), J = 1, 17816 )
END FILE 12
END FILE 12
REWRITE 12
RETURN

C 40 DECODE ( IWORD(1)* 205 )  N* ( XX(I)* I = 1, 4 )  ATIME
UNIVAC
AMN, NVIRON, ITP* ( IOP(I)* TIME(I)* I = 1, 5 )
UNIVAC
C 40 DECODE ( 80* 205* IWORD(1) )  N* ( XX(I)* I = 1, 4 )  ATIME
CDC
AMN, NVIRON, ITP* ( IOP(I)* TIME(I)* I = 1, 5 )
CDC

STIME = STIME + ATIME
IFLG = 3
RETURN
G0 IF ( IFLG .EQ. 5 )  GO TO 7D
DO 60 J = 1, 6
IX(J) = IZERO
XX(J) = BLANK
60 CONTINUE
XX(7) = BLANK
IF ( KT .EQ. IZERO .OR. IFLG .EQ. 6 )  GO TO 65
LR = 2
XX(1) = END
XX(2) = TASK
CALL RECWS
65 LR = 1
IF ( IFLG .NE. 6 )  GO TO 67
IX(1) = IFLG
CALL RECWS
67 IX(1) = 3
IX(2) = 3
IF ( KT .EQ. IZERO )  IX(2) = 1
68 CALL RECWS
70 KSAVE = KT
C DECODE ( IWORD(1)* 80 )  MN, KT* ( XX(I)* I = 1, 7 )
UNIVAC
DECODE ( 50* 80* IWORD(1) )  MN, KT* ( XX(I)* I = 1, 7 )
CDC
80 FORMAT ( A4* I3* 1X* 7A6 )
LR = 8
IX(1) = KT
DO 85 I = 2, 6
IX(I) = IZERO
85 CONTINUE
IF ( KSAVE .EQ. IZERO )  GO TO 90
IF ( STIME .LT. 1439.5 .OR. STIME .GT. 1440.5 )  CALL ERROR (9)
90 STIME = ZERO
IFLG = 2
RETURN
100 DO 110 J = 1, 6
IX(J) = IZERO
XX(J) = BLANK
110 CONTINUE
IF ( KT .EQ. IZERO .OR. IFLG .EQ. 6 )  GO TO 115
LR = 2
XX(1) = GHENOF
XX(2) = GH TASK
CALL RECWS
115 LR = 1
IF ( IFLG .NE. 6 )  GO TO 117
IX(1) = IFLG
CALL RECWS
117 IX(1) = 2
IX(2) = 3
IF ( IFLG .NE. 1 )  
GO TO 120
IX(1) = 1
IX(2) = 1
IX(4) = 1
120 CALL RECS
IF ( IFLG .NE. 1 )  
GO TO 140
LR = 6
IX(1) = 1
IX(2) = IZERO
XX(1) = 6HRUN DE
XX(2) = 6HSCRIPT
XX(3) = 6HION CA
XX(4) = 6HRD
DO 130 J = 5, 7
XX(J) = BLANK
130 CONTINUE
CALL RECS
C 140 DECODE ( IWORD(1) , 150 )                 (XX(I) , I = 1 to 7 )  
MN X KS X (XX(I) , I = 1 to 7 )  
UNIVAC
140 DECODE ( S2 , 150 , IWORD(1) )              (XX(I) , I = 1 to 7 )  
CDC
150 FORMAT ( A6 , I3 , 1X , 7A6 )               
LR = 7
IX(1) = KS
DO 160 I = 2 , 6
IX(I) = IZERO
160 CONTINUE
IFLG = 5
RETURN
200 READ (5,205) N , (XX(I) , I = 1 to 4 )  
ATIME , AMEN , NVIRON ,  
IPIER , (IOPT(I) , TIME(I) , I = 1 to 5 )  
F5 = Ge
F2.0 ~
Blr
lQXe
Ale
5
STIME = STIME + ATIME
IFLG = 3
RETURN
300 READ (5,305) (IZONE(I) , ISZONE(I) , IPRT(I) , ICODE(I) , I = 1 to 13 )  
305 FORMAT ( 26 (I2 , A1 ) )  
IJPRT = 13
DO 310 I = 1 , 13
IF ( IZONE(I) .NE. IZERO )  
GO TO 310
IF ( ISZONE(I) .NE. 1H0 .AND. ISZONE(I) .NE. 1H )  
GO TO 310
IF ( IPRT(I) .EQ. IZERO )  
GO TO 315
310 CONTINUE
GO TO 320
315 IJPRT = I - 1
320 IFLG = 4
RETURN
END
SUBROUTINE DDDD

DD AND DO CARD BUILDING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2),
APC, APS, AR(4), AREA(4:13),
BLANK, BUMP(4:11,2), FAREA(20:5,4), IAB,
ICODE(13), IDET(10), IERROR, IEX
IFLG, IJPR, INVIR(10), INSTAL,
IOP(5), IPART(11:1001), IPOINT(20:5,61), IPRT(13),
IREMOV, IRR(13), ISTART, IZONE(13),
ITYPE, IX(6), IW, IZERO,
IZONE(13), J1, JPRE, KS,
KT, M, N, NBUMP(4), NP,
NPARTS, NVIRON, N2, STIME,
TIME(5), XX(22), ZERO

DIMENSION APART(11:1001)
EQUIVALENCE ( IPART(1:1), APART(1:1) )
DIMENSION DESC(2:2), DUM(5)
EQUIVALENCE ( DESC(1:1), DUM(1) ), ( SURF, DUM(5) )
DATA ( DUM(I), I = 1, 5 ) / GH TOP, GH EXT S, GH EXT S,
6HRIOR S, 6HURFACE /

10 IF ( IJPR .LE. IZERO ) GO TO 600
DO 500 I = 1, IJPR
IF ( IRR(I) .NE. IZERO ) GO TO 500
IJ1 = IZONE(I)
IJ2 = ISUBZC ( ISZONE(I) )
JSTART = IPOINT(IJ1+IJ2-1)
IJ = 4
IF ( APART(IJ1+JSTART) .LT. ZERO ) IJ = 3
DO 10 K = 1, 6
IX(K) = IZERO
CONTINUE
10 IF ( ICODE(I) .NE. IW ) GO TO 20
JSTART = JSTART
OCCL = AMAX1 ( ( AREA(4:I) - AREA(3:I) ), ZERO )
GO TO 25
20 IJ3 = IPRT(I) + 1
JSTART = IPOINT(IJ1, IJ2, IJ3)
OCCL = AMAX1 ( ( APART(IJ1+JSTART) - APART(9, JSTART ) ), ZERO )
25 DO 400 J = 1, IJ
IF ( ICODE(I) .NE. IW ) GO TO 35
ASURF = AREA(I, J)
GO TO 50
35 ASURF = APART(J, I) + JSTART
50 IF ( J .EQ. 2 .AND. IJ .EQ. 3 ) ASURF = ASURF + OCCL
IF ( J .EQ. 4 ) ASURF = OCCL
IF ( ASURF .EQ. ZERO ) GO TO 400
IF ( ICODE(I) .EQ. IREMOV ) ASURF = ZERO
IX(I) = 16 + 4 = I + J
KJ = MIND(J+2)
M = NBUMP(KJ)
IX(2) = M
DO 100 K = 1, 2
XX(K+1) = DESC(K, KJ)
XX(K+4) = BLANK
CONTINUE
100 XX(1) = APART(5, JSTART)
XX(4) = SURF
XX(7) = BLANK
LR = 16
CALL RECWS
DO 150 K = 1 + M
XX(K) = BUMP (KJ*K+1)
XX(K+11) = BUMP (KJ*K+2) * ASURF
150 CONTINUE
KJ = M + 1
IF ( M .EQ. 11 )
60 TO 300
DO 200 K = KJ, 11
XX(K) = ZERO
XX(K+11) = ZERO
200 CONTINUE
300 XX(1) = ASURF
LR = 17
CALL RECWS
400 CONTINUE
500 CONTINUE
600 DO 700 J = 1, 6
IX(J) = IZERO
XX(J) = BLANK
700 CONTINUE
XX(7) = BLANK
LR = 16
CALL RECWS
RETURN
END
SUBROUTINE ERROR (K)

ERROR WRITING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2),
. APC, APS, AR(4), AREA(4,13)
. BLANK, BUMP(4,11,2), TAKE(20,5,4), IAB
. ICODE(13), IDET(10), ERROR, IEX
. IFLG, IJPRT, INVIR(10), INSTALL
. IOP(5), IPART(I,1001), IPINT(20,5,6), IPRT(13)
. IREMOVE, IRR(I,3), ISTART, ISZONE(I,3)
. ITYPE, IX(6), IW, IZERO
. IZONE(I,3), ILY, JPRE, KS
. KT, LH, LR, LS
. M, N, NBUMP(4), NP
. N PARTS, NVIRON, N2, STIME
. TIME(5), XX(22), ZERO

DIMENSION APART(I,1001)

EQUIVALENCE (IPART(I,1), APART(I,1))

ERROR = K
NP = NP + 1
IF (NP .GE. 55) CALL HEAD
GO TO (100, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000,
1100, 1200, 1300, 1400, 1500) K

100 WRITE (6,110) LR
110 FORMAT (56H*****ERROR***** ITP SYSTEM ERROR IN CALL TO RECWS. LR
. =* I3)
120 IF (K .EQ. 2) STOP
RETURN

200 WRITE (6,210) IPART(5,ISTART)
210 FORMAT (21H*****ERROR***** PART, A6, 25H WAS INSTALLED PREVIOUSLY
. *)
RETURN

300 WRITE (6,310) IPART(5,ISTART)
310 FORMAT (21H*****ERROR***** PART, A6, 23H WAS REMOVED PREVIOUSLY)
RETURN

400 WRITE (6,410)
410 FORMAT (32H*****ERROR***** OPERATION TIME IS ZERO)
RETURN

500 WRITE (6,510) IPART(5,ISTART)
510 FORMAT (21H*****ERROR***** PART, A4, 35H00 HAS NO AREA, NO PE CA
. POS WRITTEN)
RETURN

600 WRITE (6,610) NVIRON
610 FORMAT (53H*****ERROR***** UNRECOGNIZABLE ENVIRONMENT LETTER, CO
. DE = A1)
RETURN

700 WRITE (6,710) ISTART
710 FORMAT (55H*****ERROR***** UNRECOGNIZABLE SUBZONE LETTER, CODE =
. A1)
RETURN

800 WRITE (6,810) STIME
810 FORMAT (32H*****ERROR***** TASK TIME TOTAL: F8.2, 21H NOT EQUAL
. TO ONE DAY)
RETURN

900 WRITE (6,910) IZONE(ISTART), ISZONE(ISTART), IPRT(ISTART)
910 FORMAT (21H*****ERROR***** PART: I3, A1, I2, 18H NOT IN PARTS Li
. ST)
RETURN

1000 WRITE (6,1010) IPART(5,ISTART)
1010 FORMAT (21H*****ERROR***** PART, A6, 43H WAS NEVER INSTALLED BEF
ORE REMOVAL ATTEMPT

RETURN

1100 WRITE (6,1110)
1110 FORMAT (5DH ******ERROR***** MEN=D ON I OR R SUBTASK SET TO 1)
       AMEN = 1.
       RETURN

1200 WRITE (6,1210)  IPART(5,ISTART)
1210 FORMAT (64H ******ERROR***** OCCLUDED ALREADY BEING ADDED TO OCCLUD
       DED+ ZONE + A6 )
       RETURN

1300 WRITE (6,1310)  IPART(5,ISTART)
1310 FORMAT (63H ******ERROR***** OCCLUDED ALREADY BEING ADDED TO EXPOS
       ED+ ZONE + A6 )
       RETURN

1400 WRITE (6,1410)  XX(21), XX(22)
1410 FORMAT (40H ******ERROR***** TOTAL OPERATION TIME = * F5.0*
       * 29H NOT EQUAL TO SUBTASK TIME = * F5.0 )
       RETURN

1500 WRITE (6,1510)  IOP(ISTART)
1510 FORMAT (48H ******ERROR***** OPERATION NUMBER OUT OF RANGE = * I5*
       * 39H SET TO 1)
       IOP(ISTART) = 1
       RETURN

END
SUBROUTINE FINAL

FINAL ANALYSIS SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2), 
     APC, APS, AR(4), AREA(4*13), 
     BLANK, BUMP(4,11,2), FAREA(20,5,*), IAB, 
     ICODE(13), IDET(10), IERROR, IX, 
     IFLG, IJPRT, INVIR(10), INSTAL, 
     IOPT(5), IPART(11,1001), IPOINT(20,5,61), IPRT(13), 
     IREMOV, IRR(13), ISTART, ISZONE(13), 
     ITYPE, IX(6), IW, IZERO, 
     IZONE(13), I1, JPRE, KS, 
     KT, LK, LR, LS, 
     M, N, NBUMP(4), NP, 
     NPARTS, NVIRON, N2, STIME, 
     TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUIVALENCE (IPART(1,1), APART(1,1))

JFLG = IZERO
KFLG = IZERO
DO 500 I = 1, NPARTS
DO 100 K = 7, 10
     APART(K*I) = APART(K*I) * 144.
100 CONTINUE

DECODE (IPART(5,I), 100)  IJ1, IJ2
DECODE (6,200, IPART(5,I))  IJ1, IJ2

200 FORMAT (A4, I2)
IF (IJ2 * EQ. IZERO) GO TO 500
IF (IPART(11+I), EQ. INSTAL) GO TO 500
IF (KFLG * EQ. IZERO) WRITE (6,300)

300 FORMAT (48H1THE FOLLOWING PARTS ARE NOT CURRENTLY INSTALLED. //)
KFLG = 1
WRITE (6,400) (APART(J,I), J = 1, 11)

400 FORMAT (I4, 6A6, 4 15X, F10.0), 5X, A1)
500 CONTINUE

DO 1000 J = 1, 5
DO 900 I = 1, 20
     ISTART = IPOINT(I+J)
IF (ISTART * EQ. IZERO) GO TO 900
IF (JFLG * EQ. IZERO) WRITE (6,600)
     JFLG = 1
600 FORMAT (46H1ZONES AND THEIR FINAL AREAS, CALCULATED/INPUT. //)
     APART(10,ISTART) = APART(10,ISTART) + SIGN (APART(9,ISTART), 
     APART(10,ISTART))
     WRITE (6,700) IPART(5,ISTART), (APART(K+6,ISTART), FAREA(I+K), K = 1, 4)

700 FORMAT (15X, A6, 4 15X, F10.0, 1H/., F10.0)
IF (APART(10,ISTART) * LT. ZERO) WRITE (6,800)
800 FORMAT (65H ******ERROR****** CODE TO ADD OCLUDED TO OCLUDED NOT ENCOUNTERED)
900 CONTINUE
1000 CONTINUE
RETURN
END
SUBROUTINE HEAD

COMMON  AAG, AAS, AMEN, APA(2),
          APC, APS, AR(4), AREA(4*13),
          BLANK, BUMP(4,11,2), FAREA(20*5,4), IAB,
          ICODE(13), IDET(10), IERROR, IEX,
          IFLE, IJPR1, INVIR(10), INSTA1,
          IOP(5), IPART(11*1001), IPOINT(20*5,61), IPRT(13),
          IREM, IRR(13), ISTART, ISZONE(13),
          ITYPE, IZ(6), IW, IZERO,
          IZONE(13), I1, JPRE, KS,
          KT, LK, LR, LS,
          M, N, NBUMP(4), NP,
          NPARTS, NVIRON, N2, STIME,
          TIME(5), XX(22), ZERO

DIMENSION  APART(11,1001)
EQUIVALENCE (IPART(1,1), APART(1,1))

WRITE (6,500)
500 FORMAT (115H1RECORD, TYPE, 5/TSP/ST 1 2 3 4 5 6
          7 8 9 10 11 12 13, /)
NP = IZERO
RETURN
END
FUNCTION IOPPER (K)

FUNCTION TO DETERMINE IF OPERATION IS A DECONTAMINATION

COMMON AAG, AAS, AMEN, APA(2),
APC, APS, AR(4), AREA(4,13),
BLANK, BUMP(4,11,7), FAREA(205,4), IAB,
ICODE(13), IDET(10), IERROR, IEX,
IFLG, IPRRT, INVIR(10), INSTAL,
IOP(5), IPOINT(205,61), IPRT(13),
IREMOV, IRR(13), ISTART, ISZONE(13),
ITYPE, IX(6), IW, IZERO,
IZONE(13), I1, JPRE, KS,
KT, LK, LR, LS,
M, N, NBUMP(4), NP,
NPARTS, NVIRON, NZ, STIME,
TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUIVALENCE (IPART(1,1), APART(1,1))

IOPPER = K
DO 100 I = 1, II
IF (IDET(I) .EQ. K) GO TO 200
100 CONTINUE
RETURN
200 IOPPER = -K
RETURN
END
FUNCTION ISUBZ0 (K)

FUNCTION TO OBTAIN NUMBER FROM SUBZONE LETTER

COMMON AAG, AAS, AMEN, APA(2), APC, APS, AR(4), AREA(4*13)
* BLANK, BUMP(4,11*2), FAREA(20,5*4), IAB
* ICODE(13), IDET(10), IERROR, IEX
* IF6, IJPR1, INVR1(1L), INSTAL
* IOP(5), IPART(11,1001), IPOINT(20,5*61), IPRT(13)
* IREM0V, IRR(13), ISTART, ISZONE(13)
* ITYPE, IX(6), IW, IZERO
* IZONE(13), IL, JPRE, KS
* KT, LK, LR, LS
* M, N, NBUMP(4), NP
* NPARTS, NVIRON, N2, STIME
* TIME(5), XX(22), ZERO

DIMENSION ICHAR(5)
DATA (ICHER(1), I = 1, 5) / 1, 3, 5, 7, 9 / ISUBZ0 = IZERO
DO 100 J = 1, 5
IF (ICHER(J) .EQ. K) GO TO 200
100 CONTINUE
ISTART = K
CALL ERROR (8)
RETURN
200 ISUBZ0 = J
RETURN
END
### SUBROUTINE KC

#### KC CARD BUILDING SUBROUTINE

<table>
<thead>
<tr>
<th>COMMON AAG</th>
<th>AAS</th>
<th>AMEN</th>
<th>APA(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APC</td>
<td>APS</td>
<td>AR(4)</td>
<td>AREA(4,13)</td>
</tr>
<tr>
<td>BLANK</td>
<td>BUMP(4,11:2)</td>
<td>FAREA(20:5:4)</td>
<td>IAB</td>
</tr>
<tr>
<td>ICODE(13)</td>
<td>IDET(10)</td>
<td>IERROR</td>
<td>IEX</td>
</tr>
<tr>
<td>IF0G</td>
<td>IJ0RT</td>
<td>INVIR(10)</td>
<td>INSTAL</td>
</tr>
<tr>
<td>IOP(5)</td>
<td>IPART(11:1001),IPOINT(20:5:61),IPRT(13)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IREMOV</td>
<td>IRR(13)</td>
<td>ISTART</td>
<td>ISZONE(13)</td>
</tr>
<tr>
<td>ITYPE</td>
<td>IX(6)</td>
<td>IW</td>
<td>IZERO</td>
</tr>
<tr>
<td>IZCNE(13)</td>
<td>II</td>
<td>JPRE</td>
<td>KS</td>
</tr>
<tr>
<td>K1</td>
<td>LK</td>
<td>LR</td>
<td>LS</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td>NBUMP(4)</td>
<td>NP</td>
</tr>
<tr>
<td>NPARTS</td>
<td>NVIRON</td>
<td>NZ</td>
<td>STIME</td>
</tr>
<tr>
<td>TIME(5)</td>
<td>XX(22)</td>
<td>ZERO</td>
<td></td>
</tr>
</tbody>
</table>

**DIMENSION**
- APART(11:1001)
- EQUVALENC (IPART(1:1), APART(1:1))
- DIMENSION SIGN(2)
- DATA SIGN(1) / +1. /
- SIGN(2) / -1. /
- EQUVALENC (IX(1): KC)

**C**

| LR = 3 |
| KC = IZERO |
| CALL NVIRON |
| IF ( IJ0RT .EQ. IZERO ) GO TO 1300 |
| IKPRT = IABS ( IJ0RT ) |
| D0 1200 I = 1, IKPRT |
| IJ = IZCNE(1) |
| IF ( IJ1 .LT. 1 .OR. IJ1 .GT. 20 ) GO TO 25 |
| IJ2 = ISUB2C ( ISZONE(I) ) |
| IF ( IJ2 .EQ. IZERO ) GO TO 1200 |
| IJ3 = IPRIT(I) + 1 |
| IF ( IJ3 .LT. 1 .OR. IJ3 .GT. 61 ) GO TO 25 |
| ISTART = IPOINT(IJ1,IJ2,IJ3) |
| IF ( ISTART .NE. IZERO ) GO TO 50 |

#### 25

**ISTART = I**

| CALL ERROR (10) |
| GO TO 1200 |
| 50 JSTART = IPOINT(IJ1,IJ2,IJ3) |
| IF ( JSTART .EQ. IZERO ) GO TO 25 |
| D0 100 J = 1, 7 |
| XX(J) = ZERO |

#### 100 CONTINUE

| IX(2) = ( IJ2 - 1 ) * 20 + IJ1 |
| D0 150 J = 3, 6 |
| IX(J) = IZERO |

#### 150 CONTINUE

**KFLG = 1**

| FOR Z INSTALL-REMOVE, DELETE CARD BELOW AND ADD COMMENTED CARD |
| HOFFMA |
| C |
| IF ( ITYPE .EQ. IREMOV .AND. ( ICODE(I) .EQ. 1HZ .OR. ICODE(I) .EQ. 1W ) ) GO TO ( 200, 300 ), KFLG |
| 200 IF ( IPART(11),JSTART) .EQ. NVIRON ) GO TO 300 |
| IX(6) = NVIRON |
| IPART(11),JSTART) = NVIRON |
| 300 IF ( IRRT(I) .NE. IZERO ) GO TO 800 |
| GO TO ( 400, 500 ), KFLG |
| 400 IX(4) = 100 + 1 |
GO TO 600
500 IX(4) = ( IJ2 - 1 ) * 20 + IJ1
IX(2) = 100 + I
600 JFLG = 1
IF ( APART(10*JSTART) .LT. ZERO ) JFLG = 2
IF ( ICODE(I) .EQ. IW ) GO TO 900
DO 700 J = 1, 4
IF ( JFLG .EQ. 2 * AND. J .EQ. 2 ) GO TO 610
IF ( APART(J+6*JSTART) .EQ. ZERO ) GO TO 700
IF ( JFLG .EQ. 2 * AND. J .EQ. 4 ) GO TO 700
610 XX(1) = APART(J+6*JSTART)
IF ( J .EQ. 4 ) XX(1) = AMAX1 ( ( XX(1) - APART(9*JSTART) ),
* IX(3) = J
IX(5) = J
660 GO TO ( 690 + 675 ) * JFLG
675 IF ( J .NE. 2 ) GO TO 690
IF ( APART(10*JSTART) .EQ. ZERO ) GO TO 690
OCCL = AMAX1 ( ( APART(10*JSTART) - APART(9*JSTART) ),
* XX(1) = XX(1) + OCCL
APART(10*JSTART) = APART(10*JSTART) - OCCL = SIGNN(KFLG)
690 APART(J+6*JSTART) = APART(J+6*JSTART) + XX(1) * SIGNN(KFLG)
IF ( IX(6) .EQ. IZERO * AND. XX(1) .EQ. ZERO ) GO TO 700
IX(1) = KC + 1
CALL RECS
IX(6) = IZERO
700 CONTINUE
GO TO 1100
800 IF ( IX(6) .EQ. IZERO ) GO TO 1200
IX(11) = KC + 1
CALL RECS
GO TO 1200
900 DO 1000 J = 1, 4
IF ( JFLG .EQ. 2 * AND. J .EQ. 2 ) GO TO 910
IF ( AREA(J+1) .EQ. ZERO ) GO TO 1000
IF ( JFLG .EQ. 2 * AND. J .EQ. 4 ) GO TO 1000
910 XX(1) = AREA(J+1)
IF ( J .EQ. 4 ) XX(1) = AMAX1 ( ( XX(1) - AREA(3*I) ),
* IX(3) = J
IX(5) = J
960 GO TO ( 990 + 975 ) * JFLG
975 IF ( J .NE. 2 ) GO TO 990
IF ( AREA(4*I) .EQ. ZERO ) GO TO 990
OCCL = AMAX1 ( ( AREA(4*I) - AREA(3*I) ),
* XX(1) = XX(1) + OCCL
APART(10*JSTART) = APART(10*JSTART) - OCCL = SIGNN(KFLG)
990 APART(J+6*JSTART) = APART(J+6*JSTART) + XX(1) * SIGNN(KFLG)
IF ( IX(6) .EQ. IZERO * AND. XX(1) .EQ. ZERO ) GO TO 1000
IX(11) = KC + 1
CALL RECS
IX(6) = IZERO
1000 CONTINUE
1100 IF ( KFLG .EQ. 1 ) GO TO 1200
IF ( IPART(11*JSTART) .EQ. NVIRON ) GO TO 1200
IX(2) = ( IJ2 - 1 ) * 20 + IJ1
IX(3) = IZERO
IX(4) = IZERO
IX(5) = IZERO
IX(6) = NVIRON
XX(11) = ZERO
IX(11) = KC + 1
CALL RECS
1200 CONTINUE
1300 DO 1400 J = 1, 6
   IX(J) = IZERO
   XX(J) = ZERO
1400 CONTINUE
   XX(7) = ZERO
   CALL RECWS
   RETURN
END
SUBROUTINE KOPE
KO AND PE CARD BUILDING SUBROUTINE

COMMON AAG, AAP, AMEN, APA(2), AR(4), AREA(4,13), BAG, APS, AIR, ARE, AREA(20,5), IA, IAB, ICODE(12), IEDIT(10), IERROR, IE, IF, IH, IT, IZONE(12), I1, J, JPRE, KS, K, LK, LR, M, N, NBUMP(4), NP, NPARTS, NVIRON, N2, STIME, TIME(5), XX(22), ZERO

DIMENSION APART(11,1001), IZ(2013)

EQUIVALENCE (IPART(I), APART(I))
DIMENSION IZ(2013)

IF (AMEN .EQ. ZERO .AND. IJPRT .GT. IZERO) CALL ERROR (12)
DO 1500 I = 1, 5
DO 100 J = 1, 6
XX(J) = IZERO
XX(J) = ZERO
100 CONTINUE
XX(J) = ZERO
IF (IOP(I) .EQ. IZERO) GO TO 1600
ISTART = I
IF (IOP(I) .LT. 1 .OR. IOP(I) .GT. 35) CALL ERROR (16)
IX(1) = IOPPER (IOP(I))
IDCON = IX(1)
XX(1) = TIME(I) / 60 .0
IF (TIME(I) .EQ. ZERO) CALL ERROR (5)
IF (IDCON .LT. IZERO) GO TO 200
IX(2) = NVIRON
XX(2) = AMEN
200 CONTINUE
CALL RECWS
LR = 5
IF (IDCON .LT. IZERO) LR = 20
IF (IJPRT .EQ. IZERO) GO TO 1100
IKPRT = IABS (IJPRT)
DO 300 J = 1, IKPRT
IZ(J) = IZERO
300 CONTINUE
DO 1000 J = 1, IKPRT
IJI = IZONE(J)
IF (IJI .LT. 1 .OR. IJI .GT. 20) GO TO 325
IJ2 = ISUBZO (IZONE(JI))
IF (IJ2 .EQ. IZERO) GO TO 1000
IJ3 = IPRTR(J) + 1
IF (IJ3 .LT. 1 .OR. IJ3 .GT. 61) GO TO 325
IF (IJ3 .EQ. 1) GO TO 400
ISTART = IPPOINT(IJ1,IJ2,IJ3)
IF (ISTART .NE. IZERO) GO TO 350
325 ISTART = J
CALL ERROR (10)
GO TO 1000
350 IF (IPART(I), ISTART) .NE. INSTAL) GO TO 1000
400 ICHECK = (IJJ2 - 1) * 20 + IJ1
DO 500 K = 1, J
IF ( IZ0(J) .EQ. ICHECK ) GO TO 1000
500 CONTINUE
IZ0(J) = ICHECK
IX(1) = ICHECK
JSTART = IPOINT(IJ1+IJ2+1)
IF ( JSTART .EQ. IZERO ) GO TO 325
DO 540 K = 7, 8
IF ( APART(K+JSTART) .GT. 1.E-2 ) GO TO 560
540 CONTINUE
ISTART = JSTART
CALL ERROR (6)
GO TO 1000
560 IF ( IDCON .LT. IZERO ) GO TO 600
IX(2) = LS
XX(1) = ZERO
XX(2) = APC
XX(3) = APS
C FOR Z INSTALL-REMOVE. DELETE CARD BELOW AND ADD COMMENTED CARD
C IF ( ICODE(J) .NE. IEX .AND. ICODE(J) .NE. IEX .AND. ICODE(J) .NE. IEX )
C   1HZ ) GO TO 800
IF ( ICODE(J) .NE. IEX .AND. ICODE(J) .NE. IEX ) GO TO 800
XX(4) = APART(7+JSTART) * APA(1)
XX(5) = APART(8+JSTART) * APA(2)
GO TO 800
600 IX(2) = LK
DO 700 K = 1, 4
XX(K) = AR1(K)
700 CONTINUE
800 CALL RECWS
DO 900 K = 4, 5
XX(K) = ZERO
900 CONTINUE
1000 CONTINUE
1100 DO 1200 K = 1, 5
IX(K) = IZERO
XX(K) = ZERO
1200 CONTINUE
CALL RECWS
1500 CONTINUE
1600 DO 1700 J = 1, 5
IX(J) = IZERO
XX(J) = ZERO
1700 CONTINUE
LR = 4
CALL RECWS
RETURN
END
SUBROUTINE NVIR

SUBROUTINE TO OBTAIN ENVIRONMENT NUMBER FROM LETTER CODE

COMMON AAG, AAS, AMEN, APA(2), APC, APS, AR(4), AREA(4*13), BLANK, BUMP(4*11*2), FAREA(20*5*4), IAB, ICODE(13), IDET(10), IERROR, IEX, TFLG, IJPR, INVIR(10), INSTAL, IOP(5), IPART(11*1001), IPINT(20*5*61), IPRT(13), IREMOV, IRR(13), ISTART, ISZONE(13), ITYPE, IX(6), IZERO, IZONE(13), I1, JPRE, KS, KT, LK, LR, LS, M, N, NBUMP(4), NP, NPARTS, NVIRON, N2, STIME, TIME(5), XX(22), ZERO

DIMENSION APART(11*1001)
EQUIVALENCE (IPART(1*1), APART(1*1))

DO 100 I = 1, 10
  IF ( INVIR(I) .EQ. NVIRON .AND. NVIRON .NE. 14 ) GO TO 200
100 CONTINUE

CALL ERR(7)
NVIRON = IZERO
RETURN

200 NVIRON = I
RETURN
END
SUBROUTINE OEKC

Suproutine to build complete subtask card set for switching zone occluded areas from exposed to occluded or vice versa.

COMMON AAG, AAS, AMEN, APA(2), APC, APS, AR(4), AREA(4,13), BLANK, BUMP(4,11,2), FAREA(20,5,4), IAB, ICODE(13), IDET(10), IERROR, IEX, IFLG, IJPRT, INVIR(10), INSTAL, IOP(5), IPART(11,1G01), IPPOINT(20,5,61), IPR(13), IREMOV, IRR(13), ISTART, IZONE(13), ITYPE, IX(6), IZ(13), IL1, JPRT, KS, KT, L, LK, LR, LS, M, N, NBUMP(4), NP, NPARTS, NVIRON, N2, STIME, TIME(5), XX(22), ZERO

DIMENSION APART(11,1G01)

EQUIVALENCE (IPART(1,1), APART(1,1))

EQUIVALENCE (IX(1), KC)

DIMENSION IZO(13)

DATA IO / 1HO /

IE / 1HE /

KC = IZERO
JFLG = IZERO
IKPRT = IABS(IJPRT)
DO 100 IO = 1, IKPRT
IZO(1) = IZERO
IF (ICODE(I) .NE. IC .AND. ICODE(I) .NE. IE) GO TO 100
JFLG = JFLG + 1
IZO(JFLG) = I

100 CONTINUE
IF (JFLG .EQ. IZERO) RETURN

IX(1) = N
XX(1) = CHCHANGE
XX(2) = GH 0 + E
XX(3) = GH ZONE
XX(4) = GHAREAS
DO 200 IO = 5, 7
XX(I) = BLANK

200 CONTINUE
LR = 2
CALL RECWS
DO 300 IO = 1, 4
XX(I) = BLANK

300 CONTINUE
IX(1) = IZERO
LR = 14
CALL RECWS
LR = 16
CALL RECWS
DO 350 IO = 1, 7
XX(I) = ZERO

350 CONTINUE
IX(16) = IZERO
LR = 3
DO 1000 IO = 1, JFLG
IJ = IZO(I)
IJ1 = IZONE(IJ)
IF (IJ1 .LT. 1 .OR. IJ1 .GT. 20) GO TO 375
I J2 = I SUBZ0( ISZONE(I J1 )
IF ( I J2 .EQ. I ZERO ) GO TO 1000
ISTART = IPOINT( I J1 + I J2 +1)
IF ( I START .NE. I ZERO ) GO TO 400
375 I START = I J
CALL EP ROR (10)
GO TO 1000
400 I X(2) = ( I J2 - 1 ) * 20 + I J1
I X(4) = I X(2)
XX(1) = ABS( APART(10*ISTART ) )
IF ( XX(1) .LE. 1. E-2 ) XX(1) = ZERO
KFLG = 1
IF ( I CODE( I J ) .EQ. IE ) KFLG = 2
GO TO ( 500, 700 ) KFLG
500 I X(3) = 4
I X(5) = 2
XX(1) = MIN1( XX(1) * APART(8*ISTART ) )
IF ( APART(10+ISTART) .LT. ZERO ) GO TO 600
CALL EP ROR (13)
GO TO 1000
600 APART(8+ISTART) = APART(8+ISTART) - XX(1)
APART(10+ISTART) = XX(1)
GO TO 900
700 I X(3) = 2
I X(5) = 4
IF ( APART(10+ISTART) .GE. ZERO ) GO TO 800
CALL ERROR (14)
GO TO 1000
800 APART(8+ISTART) = APART(8+ISTART) + XX(1)
APART(10+ISTART) = MIN1( (-XX(1)) * (-5.E-3) )
900 I X(1) = K C + 1
CALL PECWS
1000 CONTINUE
DO 1100 I = 1, 5
I X(I) = I ZERO
1100 CONTINUE
XX(1) = ZERO
CALL PECWS
LR = 4
CALL PECWS
RETURN
END
SUBROUTINE POINTL (KOUNT)

PD AND PG CARD BUILDING SUBROUTINE FOR ZONES (PARTS 1 TO 100)

COMMON AAQ, AAS, AMEN, APA(2),
APC, APS, AR(4), AREA(40,13),
BLANK, BUMP(40,11,2), FAREA(20,5,4), IAB,
ICODE(13), IDET(10), IERROR, IEX,
IFLG, IPRT, INVIR(10), INSTAL,
IOP(5), IPART(11,100), IPPOINT(20,5,61), IPRT(13),
IREMOV, IRR(13), ISTART, ISZONE(13),
ITYPE, IX(6), IW, IZERO,
IZONE(13), K, JPRE, KS,
K, LK, LR, LS,
M, N, NBUMP(4), NP,
NPART, NVIRON, N2, STIME,
TIME(5), XX(22), ZERO

DIMENSION APART(11,100)

EQUIVALENCE (IPART(1,1), APART(1,1))

KOUNT = 1
DO 400 J = 1, 5
DO 300 I = 1, 20
ISTART = IPPOINT(I,J,1)
IF (ISTART.EQ. IZERO) GO TO 300
DO 100 K = 1, 6
IX(K) = IZERO
XX(K) = APART(K,ISTART)
100 CONTINUE
XX(7) = BLANK
NZONE = (J - 1) * 20 + I
IX(1) = NZONE
LR = 14
CALL RECWS
DO 200 K = 1, 4
IX(K) = 96 + 4 * NZONE + K
XX(K+2) = ZERO
200 CONTINUE
XX(7) = ZERO
IX(6) = IAB
XX(1) = AAQ
XX(2) = AAS
LR = 15
CALL RECWS
300 CONTINUE
400 CONTINUE
RETURN
END
SUBROUTINE PDPO

PD AND PD CARD BUILDING SUBROUTINE

COMMON AAG, AAS, AMEN, APA(2),
      APC, APS, AR(4), AREA(4,13),
      BLANK, BUMP(4,11,2), FAREA(20,5,14), IAB,
      ICODE(13), IDET(10), IERROR, IEX,
      IFLG, JPRT, INVIR(10), INSTAL,
      IOP(15), IPART(11,1001), IPOINT(20,5,61), IPRT(13),
      IREMOV, IRR(13), ISTART, ISZONE(13),
      ITYPE, IX(6), IW, IZERO,
      IZONE(13), I1, JPRE, KS,
      KT, LK, LR, LS,
      M, N, Nbump(4), NP,
      NPARTS, NVIRON, N2, STIME,
      TIME(5), XX(22), ZERO

DIMENSION APART(11,1001)
EQUIVALENCE (IPART(1,1), APART(1,1))

DO 15 I = 1, IJPRT
  IRR(I) = 1
  15 CONTINUE

IF ( ITYPE .NE. INSTAL .AND. ITYPE .NE. IREMOV ) IJPRT = -IJPRT
IF ( IJPRT .LE. IZERO ) GO TO 1800
DO 1700 I = 1, IJPRT
  IJ1 = IZONE(I)
  IF ( IJ1 .LT. 1 .OR. IJ1 .GT. 20 ) GO TO 20
  IJ2 = ISUB2C (ISZONE(I))
  IF ( IJ2 .EQ. IZERO ) GO TO 1700
  IJSTART = IPOINTER(IJ1, IJ2, 1)
  IF ( IJSTART .EQ. IZERO ) GO TO 20
  IJ = 4
  IF ( APART(IJ+IJSTART) .LT. ZERO ) IJ = 3
  DO 20 IJ = 1, IJPRT
    IF ( ICODE(I) .NE. IHZ ) GO TO 500
    IF ( ICODE(I) .NE. IEX ) GO TO 500
    IJ3 = IPRT(I) + 1
    IF ( IJ3 .LT. 1 .OR. IJ3 .GT. 61 ) GO TO 20
    IF ( IJ3 .EQ. 1 ) GO TO 1700
    IJSTART = IPOINTER(IJ1, IJ2, IJ3)
    IF ( IJSTART .NE. IZERO ) GO TO 25
  20 SSTART = 1
  CALL ERROR (10)
  GO TO 1700

25 IERROR = IZERO
  IF ( ITYPE .EQ. IPART(11, ISTART) .AND. ITYPE .EQ. INSTAL )
    CALL ERROR (3)
  IF ( ITYPE .EQ. IPART(11, ISTART) .AND. ITYPE .EQ. IREMOV )
    CALL ERROR (4)
  IF ( ITYPE .EQ. IREMOV .AND. IPART(11, ISTART) .EQ. IHN )
    CALL ERROR (11)
  IRR(I) = IERROR
  IF ( IERROR .NE. IZERO ) GO TO 1700
  IPART(11, ISTART) = ITYPE
  DO 50 J = 7, 10
    IF ( APART(J, ISTART) .NE. ZERO ) GO TO 75
  50 CONTINUE
  CALL ERROR (6)
  IRR(I) = 1
  GO TO 1700
75 DO 100 J = 1, 6
   IX(J) = IZERO
   XX(J) = APART(J, ISTART)
100 CONTINUE
   XX(7) = BLANK
   IX(1) = 100 * I
   LR = 14
   CALL RECWS
   DO 200 J = 2, 7
      XX(J) = ZERO
200 CONTINUE
   IX(1) = IZERO
   IX(6) = IAB
   XX(1) = AAG
   XX(2) = AAS
   LR = 15
   OCCL = AMAX1 ( (APART(10, ISTART) - APART(9, ISTART)) * ZERO )
   DO 300 J = 1, IQ
      IX(J) = 16 + 4 * I + J
      ASURF = APART(J, ISTART)
      IF ( J, EQ, 2 )  AND. J, EQ, 3 )  ASURF = ASURF + OCCL
      IF ( J, EQ, 4 )  ASURF = OCCL
      IF ( ASURF, EQ, ZERO )
         IX(J) = IZERO
300 CONTINUE
   CALL RECWS
   C500 IF ( ICODE[1], NE, IW )  GO TO 1700
   C   DECODE ( IPART(S, JSTART), 600 )  ICODE1
   C   DECODE ( 4, EGC, IPART(S, JSTART) )  ICODE1
   600 FORMAT ( A4 )
   DO 700 K = 1, 4
      AREA(K+1) = ZERO
700 CONTINUE
   C   DO 900 J = 1, 60
      ISTART = JSTART + J
   C   DECODE ( 4, EGC, IPART(S, ISTART) )  ICODE2
   C   DECODE ( 4, 600, IPART(S, ISTART) )  ICODE2
   C   IF ( ICODE2, NE, ICODE1 )  GO TO 1000
   C   IERROR = IZERO
   C   IF ( ITYPE, EQ, IPART(11, ISTART) )  AND. ITYPE, EQ, INSTAL
      CALL ERROR (3)
   C   IF ( ITYPE, EQ, IPART(11, ISTART) )  AND. ITYPE, EQ, IREM0V
      CALL ERROR (4)
   C   IF ( ITYPE, EQ, IREM0V )  AND, IPART(11, ISTART) )  EQ, 1HN
      CALL ERROR (11)
   C   IF ( IERROR, NE, IZERO )  GO TO 900
   C   IPART(11, ISTART) = ITYPE
   DO 800 K = 1, 4
      AREA(K+1) = AREA(K+1) + APART(K+6, ISTART)
800 CONTINUE
   800 CONTINUE
   900 CONTINUE
   C1000 DO 1100 K = 1, 4
      IF ( AREA(K+1), NE, ZERO )  GO TO 1200
1100 CONTINUE
   GO TO 1700
   C1200 IRR(J) = IZERO
   DO 1300 J = 1, 6
      IX(J) = IZERO
   CONTINUE
XX(J) = APART(J+JSTART)
1300 CONTINUE
XX(7) = BLANK
IX(1) = 1CD * I
LR = 14
CALL RECWS
C
DO 1400 J = 2 * 7
XX(J) = ZERO
1400 CONTINUE
IX(1) = IZERO
IX(6) = IAB
XX(1) = AAG
XX(2) = AAS
LR = 15
C
OCCL = AMAX1 ( ( AREA(4,I) - AREA(3,I) ) , ZERO )
DO 1650 J = 1 * IJ
IX(J) = 16 + 4 * I * J
ASURF = AREA(J*I)
IF ( J .EQ. 2 .AND. IJ .EQ. 3 ) ASURF = ASURF + OCCL
IF ( J .EQ. 4 ) ASURF = OCCL
IF ( ASURF .EQ. ZERO ) IX(J) = IZERO
1650 CONTINUE
CALL RECWS
1700 CONTINUE
C
1800 DO 1900 J = 1 * 6
IX(J) = IZERO
XX(J) = BLANK
1900 CONTINUE
XX(7) = BLANK
LR = 14
CALL RECWS
RETURN
END
SUBROUTINE PNPUT

PRELIMINARY INPUT SUBROUTINE

COMMON  AAG  ,  AAS  ,  AMEN  ,  APA(2)  ,
         APC  ,  APS  ,  AR(4)  ,  AREA(4*13)  ,
         BLANK  ,  BUMP(4*11*2)  ,  FAREA(20*5*4)  ,  IMAB  ,
         ICODE(13)  ,  IDET(13)  ,  IERROR  ,  IEX  ,
         IFLG  ,  IJPRJ  ,  INVIR(10)  ,  INSTAL  ,
         IOP(5)  ,  IPART(11*1001)  ,  IPOINT(20*5*61)  ,  IPRT(13)  ,
         IREMOV  ,  IRR(13)  ,  ISTART  ,  ISZONE(13)  ,
         ITYPE  ,  IX(6)  ,  IW  ,  IZERO  ,
         IZONE(13)  ,  II  ,  JPRE  ,  KS  ,
         KT  ,  LM  ,  LR  ,  LS  ,
         M  ,  N  ,  NBUMP(4)  ,  NP  ,
         NPARTS  ,  NVIRON  ,  N2  ,  STIME  ,
         TIME(5)  ,  XX(22)  ,  ZERO

DIMENSION  APART(11*1001)
EQUIVALENCE  ( IPART(11*1)  ,  APART(11*1) )

KS = 1
NP = 60
ZERO = C.
IZERO = D
KT = IZERO
N2 = IZERO
IFLG = IZERO
IERROR = IZERO
STIME = ZERO
BLANK = EH
IBLANK = EH
INSTAL = IH
IREMOV = IHR
IEX = IHX
IW = IHW
DO 75  I = 1, 20
DO 50  J = 1, 5
DO 25  K = 1, 61
IPCI INT(I,J,K) = IZERO
25 CONTINUE
50 CONTINUE
75 CONTINUE
DO 100  I = 1, 1001
DO 95  J = 1, 10
IPART(J,I) = IZERO
95 CONTINUE
IPART(11*I) = 1HN
100 CONTINUE
DO 130  I = 1, 1001
READ (5,105)  IPART(5*I), (IPART(J*I), J = 1, 4)
( APART(J*I), J = 7, 10 )  IPART(6*I)
105 FORMAT ( A6, 3X, 4A6, 1X, 4F9.0, 4X, A6 )
IF ( IPART(5*I) .EQ.  IBLANK )  GO TO 140
C DECODE ( IPART(5*I) )  ICODE1, ICODE2, ICODE3
DECODE ( E+ 11C  IPART(5*I) )  ICODE1, ICODE2, ICODE3
110 FORMAT ( I3, A1, I2 )
K = ISUB20 (ICODE2)
IF ( K .NE. IZERO )  GO TO 125
115 WRITE (6+12C)
120 FORMAT ( 43H1*****ERROR***** ZONE NOT IN RANGE 1 TO 20,*)
     / / 17X 31HSUBZONE NOT IN RANGE A TO E, OR,,
     / / 17X 25HPART NOT IN RANGE 0 TO 60, // )
WRITE (6,105) IPART(5*I), IPART(J*I), J = 1, 4, I

, IERROR = IERROR + 1
GO TO 130

125 ICODE2 = K
IF ( IC0DE1 .LT. 1 .OR. IC0DE1 .GT. 10 ) GO TO 115
IF ( IC0DE3 .LT. IZERO .OR. IC0DE3 .GT. 60 ) GO TO 115
IC0DE3 = IC0DE3 + 1
IPOINT(IC0DE1, IC0DE2, IC0DE3) = I

130 CONTINUE
WRITE (6,135)

135 FORMAT ( 5CH1=****ERROR**** MORE THAN 1000 PARTS IN PART LIST,
, 17H OR NO TERMINATOR; /* 17X, 17H UNABLE TO PROCEED )
STOP

140 M0RTS = I - 1
IF ( IERROR .EQ. IZERO ) GO TO 150
IFLG = IFLG + 1
WRITE (6,145) IERROR

145 FORMAT ( // 110, 42H ZONE, SUBZONE, AND PART IDENTIFIER ERRORS,
, 9H DETECTED, / )
IERROR = IZERO

150 IF ( IFLG .NE. IZERO ) GO TO 205
DO 165 I = 1, 20
DO 165 J = 1, 5
ISTART = IPOINT(I*J, I)
IF ( ISTART .EQ. IZERO ) GO TO 160
IPART(I*J*ISTART) = IZERO
DO 165 K = 1, 4
FAREA(I*J*K) = APART(K*6+ISTART)
APART(K*6+ISTART) = ZERO

155 CONTINUE
APART(1C*ISTART) = -72

160 CONTINUE

165 CONTINUE

205 DO 230 I = 1, 2
NBUMP(I) = IZERO
READ (5,210) NBUMP(I)

210 FORMAT ( 110 )
IF ( NBUMP(I) .LT. 1 .OR. NBUMP(I) .GT. 11 ) GO TO 220
IBUMP = NBUMP(I)
READ (5,215) ( BUMP(I*J+1), J = 1, IBUMP )
READ (5,215) ( BUMP(I*J+2), J = 1, IBUMP )

215 FORMAT ( 8E10.3 )
GO TO 230

220 WRITE (6,225)

225 FORMAT ( 51H1=****ERROR**** NO. OF VALUES NOT IN RANGE 1 TO 11,
, // )
WRITE (6,210) NBUMP(I)
IERROR = IERROR + 1
READ (5,215) DUMMY
READ (5,215) DUMMY

230 CONTINUE

240 IF ( IERROR .EQ. IZERO ) GO TO 250
IFLG = IFLG + 1
WRITE (6,245) IFLG

245 FORMAT ( 16H1=ERRORS EXIST IN, 12, 27H GROUPS OF INPUT; THEREFORE,
, // 8X, 22HPROGRAM IS TERMINATING )
STOP

250 READ (5,260) IAB, AAG, AAS, LS, APC, APS ( APA(K), K = 1, 2 )
, LK ( ARI(K), K = 1, 4 )

260 FORMAT ( I10, 2E10.3, I10, 4E10.3, // I10, 4E10.3 )
READ (5,270) I10 ( IDET(I), I = 1, II )
270 FORMAT ( 11I5 )
    DO 270 I = 1, NPARTS
    DO 275 J = 7, 10
         APART(J,I) = APART(J,I) / 144.
275 CONTINUE
280 CONTINUE
    READ (5,280) ( INVIR(K), K = 1, 10 )
290 FORMAT ( 10 ( 1X, A1 ) )
    IFLG = 1
    RETURN
END
SUBROUTINE RECWS

RECORD WRITING SUBROUTINE, LR IS CURRENT RECORD TYPE

COMMON AAG, AAS, AMEN, APA(2),
          APC, APS, AR(4), AREA(4+13),
          BLANK, BUMP(4+11,2), FAREA(20,5+4), IAB,
          ICODE(13), IDET(10), IERROR, IEK,
          IFLG, JPRT, INVR(10), INSTAL,
          IOP(5), IPART(11,1001), IPOINT(20,5,61), IPRT(13),
          IREMVT, IRR(13), ISTART, ISZONE(13),
          IZONE(13), IN, IZERO,
          KT, LK, LR, LS,
          M, N, NBUMP(4), NPE,
          NPARTS, NVIRON, N2, STIME,
          TIME(5), XX(22), ZERO

DIMENSION APART(11,1001),
       EQUIVALENCE ( IPART(11,1), APART(11,1) )

DIMENSION CT(21)
DATA ( CT(I), I = 1, 21 ) / 2HCC, 2HKD, 2HKC, 2HK0, 2HPE, 2HRD,
       2HS0, 2HTD, 2HE0, 2HEO, 2H0, 2HOQ, 2HPD, 2HPQ,
       2HD0, 2HDQ, 2HZC, 2HPE, 2HER /
DATA KOUNT / 0 /

FORMAT ( I7, 3X, A2, I4, 1H/* I3, 1H/* I2 )
N2 = N2 + 1
NP = NP + 1
IF ( LR .EQ. 17 )          NP = NP + 2
IF ( NP .GE. 55 )          CALL HEAD
GO TO ( 50, 100, 200, 300, 400, 450, 500, 600, 1500, 1500, 1500, 1500, 1500, 1500, 1500, 1100, 1200, 1300 )

WRITE (6,601) N2, CT(LR)
WRITE (6,613) IX
613 FORMAT ( 1H/*, 23X, 6I4 )

GO TO 1300

WRITE (6,601) N2, CT(LR), KS, KT, N
WRITE (6,602) ( IX(J)* J = 1, 3 ) ( XX(J)* J = 1, 4 )
602 FORMAT ( 1H/*, 23X, 3I4, 17X, 4A6 )

GO TO 1300

WRITE (6,601) N2, CT(LR), KS, KT, N
WRITE (6,603) IX, ( XX(J)* J = 1, 3 )
603 FORMAT ( 1H/*, 23X, 3I4, 3F10.3 )

GO TO 1300

WRITE (6,601) N2, CT(LR), KS, KT, N
WRITE (6,604) ( IX(J)* J = 1, 2 ) ( XX(J)* J = 1, 2 )
604 FORMAT ( 1H/*, 23X, 2I4, 16X, 2F10.3 )

GO TO 1300

WRITE (6,601) N2, CT(LR), KS, KT, N
WRITE (6,605) ( IX(J)* J = 1, 2 ) ( XX(J)* J = 1, 5 )
605 FORMAT ( 1H/*, 23X, 2I4, 16X, 5F10.3 )

GO TO 1300

WRITE (6,614) IX(1)* ( XX(J)* J = 1, 7 )
614 FORMAT ( 1H40, 23X*, I4*, 25X*, 7AE )
GO TO 1300
C
500 WRITE (6,601) N2*, CT(LR)*, KS
WRITE (6,606) KS*, ( XX(J)*, J = 1*7 )
606 FORMAT ( 1H40, 23X*, I4*, 25X*, 7AE )
GO TO 1300
C
600 WRITE (6,601) N2*, CT(LR)*, KS*, KT
WRITE (6,607) ( IX(J)*, J = 1*3 ), ( XX(J)*, J = 1*7 )
607 FORMAT ( 1H40, 23X*, 3I4*, 17X*, 7AE )
GO TO 1300
C
700 WRITE (6,601) N2*, CT(LR)*, KS*, KT* N
WRITE (6,608) IX(1), ( XX(J)*, J = 1*6 )
608 FORMAT ( 1H40, 23X*, I4*, 25X*, 6A6 )
GO TO 1300
C
800 WRITE (6,601) N2*, CT(LR)*, KS*, KT* N
WRITE (6,609) IX*, ( XX(J)*, J = 1*2 )
609 FORMAT ( 1H40, 23X*, I4*, 2F10.3 )
GO TO 1300
C
900 WRITE (6,601) N2*, CT(LR)*, KS*, KT* N
WRITE (6,610) IX*, ( XX(J)*, J = 1*4 )
610 FORMAT ( 1H40, 23X*, 2I4*, 2I4*, 21X*, 4A6 )
GO TO 1300
C
1000 WRITE (6,601) N2*, CT(LR)*, KS*, KT* N
WRITE (6,611) ( XX(J)*, J = 1* M )
611 FORMAT ( 8X*, F10.2, 10F10.6 )
MP = M * 11
WRITE (6,612) ( XX(J)*, J = 12*, MP )
612 FORMAT ( 8X*, 11E10.2 )
GO TO 1400
C
1100 WRITE (6,601) N2*, CT(LP)*, KS*, KT* N
WRITE (6,605) ( IX(J)*, J = 1*2 ), ( XX(J)*, J = 1*4 )
GO TO 1300
1200 CONTINUE
1300 WRITE (12) LR*, IX*, ( XX(J)*, J = 1*7 )
RETURN
1400 WRITE (12) LR*, XX
RETURN
1500 N2 = N2 - 1
NP = NP - 1
KOUNT = KCOUNT + 1
IF ( KCOUNT .EQ. 20 )
CALL ERROR (2)
CALL ERROR (1)
RETURN
END