THERMAL AND FLOW ANALYSIS SUBROUTINES
FOR THE SINDA-VERSION 9 COMPUTER ROUTINE

REPORT NO. 00.1582

24 September 1973

(NASA-CR-134121) THERMAL AND FLOW
ANALYSIS SUBROUTINES FOR THE SINDA-VERSION
9 COMPUTER ROUTINE (LTV Aerospace Corp.)
274 p HC $15.75

CSCL 20D

VOUGHT SYSTEMS DIVISION
LTV AEROSPACE CORPORATION
PERFORMED UNDER
NASA-JSC CONTRACT
NAS9-6807

THERMAL AND FLOW ANALYSIS SUBROUTINES
FOR THE SINDA-VERSION 9 COMPUTER ROUTINE

REPORT NO. 00.1582
24 SEPTEMBER 1973

SUBMITTED BY
VOUGHT SYSTEMS DIVISION
LTV AEROSPACE CORPORATION
P.O. BOX 5907
DALLAS, TEXAS

TO
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

PREPARED BY:

D. A. OREN
D. R. WILLIAMS

APPROVED BY:
R. J. FRENCH, SUPERVISOR
ENVIRONMENTAL CONTROL/
LIFE SUPPORT SYSTEMS
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 SUMMARY AND INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2.0 DISCUSSION OF METHODS</td>
<td>2</td>
</tr>
<tr>
<td>2.1 Thermal Analysis Features</td>
<td>2</td>
</tr>
<tr>
<td>2.1.1 Convection Conductors</td>
<td>2</td>
</tr>
<tr>
<td>2.1.2 Flow Conductors</td>
<td>4</td>
</tr>
<tr>
<td>2.1.3 Heat Exchanger Analysis</td>
<td>5</td>
</tr>
<tr>
<td>2.1.4 Inline Heater Analysis</td>
<td>8</td>
</tr>
<tr>
<td>2.1.5 Cabin Analysis</td>
<td>8</td>
</tr>
<tr>
<td>2.1.6 Radiation Interchange Analysis</td>
<td>14</td>
</tr>
<tr>
<td>2.2 Fluid Flow Analysis</td>
<td>25</td>
</tr>
<tr>
<td>2.2.1 Overall Flow Model Description</td>
<td>25</td>
</tr>
<tr>
<td>2.2.2 Tube Conductor Determination</td>
<td>29</td>
</tr>
<tr>
<td>2.2.3 Valve Analysis</td>
<td>31</td>
</tr>
<tr>
<td>2.2.4 Pressure-Flow Network Solution</td>
<td>37</td>
</tr>
<tr>
<td>2.2.5 Pump and System Pressure-Flow Matching</td>
<td>40</td>
</tr>
<tr>
<td>3.0 MODIFICATIONS TO SINDA SUBROUTINES</td>
<td>45</td>
</tr>
<tr>
<td>3.1 Preprocessor Modifications</td>
<td>45</td>
</tr>
<tr>
<td>3.2 Execution Routine Modifications</td>
<td>45</td>
</tr>
<tr>
<td>4.0 USER SUBROUTINES</td>
<td>46</td>
</tr>
<tr>
<td>5.0 SAMPLE PROBLEM</td>
<td>91</td>
</tr>
<tr>
<td>6.0 REFERENCES</td>
<td>94</td>
</tr>
</tbody>
</table>

## APPENDICES

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>SUBROUTINE LISTINGS</td>
<td>A-1</td>
</tr>
<tr>
<td>B</td>
<td>USERS DESCRIPTION FOR PLOTA</td>
<td>B-1</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>Illustration of Method Used to Determine Specular Surface Reflected View Factors</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>Flow System Schematic</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>Main Network and Subnetworks</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>Friction Factor vs Reynolds Number</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>Rate Limited Valve Operation</td>
<td>34</td>
</tr>
<tr>
<td>6</td>
<td>System/Pump Curve Solution</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>Flow Charts of PFCS and NTWRK</td>
<td>80</td>
</tr>
<tr>
<td>8</td>
<td>Flow Charts of NTWRK1 and NTWRKN</td>
<td>81</td>
</tr>
<tr>
<td>9</td>
<td>Flow Chart of FLOBAL</td>
<td>82</td>
</tr>
<tr>
<td>10</td>
<td>Fluid Model of the Sample Problem</td>
<td>92</td>
</tr>
<tr>
<td>11</td>
<td>Structure Model for the Sample Problem</td>
<td>93</td>
</tr>
<tr>
<td>12</td>
<td>Radiator Temperatures For Sample Problem</td>
<td>148</td>
</tr>
<tr>
<td>13</td>
<td>System Temperatures For Sample Problem</td>
<td>149</td>
</tr>
<tr>
<td>14</td>
<td>System Pressures For Sample Problem</td>
<td>150</td>
</tr>
<tr>
<td>15</td>
<td>Radiator Flow Rates For Sample Problem</td>
<td>151</td>
</tr>
<tr>
<td>16</td>
<td>Pump Pressures For Sample Problem</td>
<td>152</td>
</tr>
<tr>
<td>17</td>
<td>System Pressures For Sample Problem</td>
<td>153</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>New SINDA User Subroutines</td>
<td>47</td>
</tr>
<tr>
<td>2</td>
<td>Value of GC For Various Problem Units</td>
<td>79</td>
</tr>
<tr>
<td>3</td>
<td>Output For Sample Problem</td>
<td>94</td>
</tr>
<tr>
<td>B-1</td>
<td>Correspondence Between Fortran Unit No. &amp; I/O Device</td>
<td>B-9</td>
</tr>
</tbody>
</table>
1.0 SUMMARY AND INTRODUCTION

During the past decade extensive capabilities for combined thermal and fluid flow transient analysis was developed at the Vought Systems Division (VSD) of LTV Aerospace Corporation. The capabilities included (1) a pressure/flow solution for a general flow network (integrated with the finite difference temperature solution) including general valve analyses, orifice and pump analysis packages, (2) a number of special thermal analysis options including heat exchanger analysis, cavity radiant interchange analysis, cabin analysis, etc. and (3) a number of input/output capabilities such as automatic plotting, interrupt and restart, etc. These capabilities were included in a general purpose thermal analysis routine, MOTAR\textsuperscript{1}, developed by VSD for NASA-JSC.

The objective of the effort described by this report was to incorporate these fluid flow analysis, special thermal analysis and input/output capabilities of the MOTAR routine into the SINDA\textsuperscript{7} routine which was developed by the TRW Corporation. This effort was performed under contract NAS9-6807 for NASA-JSC. All the capabilities were added in the form of user subroutines so that they may be added to different versions of SINDA with a minimum of programmer effort.

Two modifications were made to the existing subroutines of SINDA/VERSION 8 to incorporate the above subroutines. These were:

1. A modification to the preprocessor to permit actual values of array numbers, conductor numbers, node numbers or constant numbers supplied as array data to be converted to relative numbers.

2. Modifications to execution subroutine CNFAST to make it compatible with the radiant interchange user subroutine, RADIR.

This modified version of SINDA has been designated SINDA/VERSION 9.

A detailed discussion of the methods used for the capabilities added is presented in Section 2.0. The modifications for the SINDA subroutines are described in Section 3.0. User subroutines are described in Section 4.0, and a sample problem is given in Section 5.0. All subroutines added or modified are listed in Appendix A.

\textsuperscript{*} Superscripts refer to references in Section 6.0
2.0 DISCUSSION OF METHODS

SINDA user subroutines were developed to incorporate the MOTAR routine's capabilities for fluid/pressure analysis, thermal analysis of a flowing fluid and enclosure analysis into the SINDA routine. The analytical methods for these capabilities are described in this section. Thermal analysis features such as those methods required for analysis of a flowing fluid and those required for enclosure radiation analysis are described in Section 2.1. Pressure-flow analysis methods are described in Section 2.2.

2.1 Thermal Analysis Features

The calculation methods for (1) convection and flow thermal conductors for flow in a tube, (2) heat exchanger thermal performance, (3) inline heater thermal performance, (4) cabin thermal and mass balance, and (5) enclosure radiation thermal performance have been added to the SINDA library of user subroutines. The methods used are based on those from the MOTAR computer routine and are described in detail in the following sections.

2.1.1 Convection Conductors

Three user subroutines were prepared for the SINDA library to give the user the capability of analyzing convection heat transfer for flow in a tube. These subroutines and their functions are:

CONV1 - Calculates heat transfer coefficient using relationships for convection in a flowing tube
CONV2 - Calculates the heat transfer coefficient using the Stanton number obtained from interpolating a curve
CONV3 - Interpolates a curve of heat transfer coefficients vs flowrates for the coefficient

The value of the convection conductor, \( G_{ij} \), between a fluid lump and tube lump is given by the following relation for all three of the above routines:

\[
G_{ij} = hA
\]  

(1)

where

- \( h \) - the convection heat transfer coefficient
- \( A \) - the convection area
CONVI uses one of several methods for determining the heat transfer coefficient, \( h \), for flowing fluid in a tube depending on the flow regime. The flow regime is assumed to be laminar when the Reynolds number is 2000 or less. For this regime the convection heat transfer coefficient is calculated by:

\[
h = \frac{k}{D} \left[ 3.66 \cdot F_1 + \frac{0.0155 \cdot F_2}{\frac{1}{Re Pr} \frac{X}{D} + 0.015 \frac{1}{Re Pr} \left( \frac{X}{D} \right)^{1/3}} \right]
\]

where:
- \( k \) = thermal conductivity
- \( D \) = hydraulic diameter to flow
- \( X \) = distance from tube entrance
- \( Re \) = Reynolds number
  \[ Re = \frac{4 \cdot \dot{m} \cdot \mu}{\rho \cdot P} \]
- \( \dot{m} \) = flow rate of fluid
- \( \mu \) = viscosity of fluid
- \( P \) = wetted perimeter of fluid flow passage
- \( F_1 \) = An input factor for modifying fully developed flow
- \( F_2 \) = An input factor for modifying developing flow

Equation (2) is a curve fit obtained by VMSC to approximate the Gratz solution\(^2\) to flow in a tube for values of \( \frac{X}{D} \cdot \frac{1}{Re Pr} \) greater than 0.001.

The convection heat transfer coefficient for flow in a tube in the transition flow regime (2000 < \( Re \) < 6400) is approximated in CONVI by the following relation:

\[
h = \frac{K}{D} \left[ 0.116 \left( Re^{2/3} - 125(Pr)^{1/3} \right) \right]
\]

This relation was derived by Hausen\(^2\) and holds only for fully developed flow.
The relation used in CONV1 to determine $h$ for turbulent flow ($Re > 6400$) is the following:

$$h = 0.023 K D (Re) B(Pr)^{1/3}$$  \hspace{1cm} (4)$$

CONV2 supplies a more general option for determining the convection heat transfer coefficient. A curve of $St(Pr)^{2/3}$ vs Reynolds No. is interpolated to obtain the value of $St(Pr)^{2/3}$. That is,

$$St(Pr)^{2/3} = F(Re)$$  \hspace{1cm} (5)$$

Where $St = Stantion number$

$$= \frac{Nu}{Re Pr}$$

$$= \frac{h}{CpV}$$

$V$ = Average fluid velocity

$F(Re)$ = An arbitrary function of Reynolds number which the user can input as a table

The heat transfer coefficient is calculated by

$$h = K \frac{F(Re)}{D} Re(Pr)^{1/3}$$  \hspace{1cm} (6)$$

In CONV3, the convection heat transfer coefficient is obtained by direct interpolation of a curve of heat transfer coefficient vs flowrate.

2.1.2 Flow Conductors

A method for calculating the value of flow conductors is required when analyzing a problem with fluid flowing in a tube. The flow conductor is a one way conductor from node $i$ to node $j$ and is calculated by

$$G_{ij} = \dot{W} C_{pi}$$  \hspace{1cm} (7)$$

where: $G_{ij} =$ the conductance from the upstream lump

$\dot{W} =$ the mass flow rate in the tube

$C_{pi} =$ the fluid specific heat for lump $i$
Two user subroutines FLOCN1 and FLOCN2, were prepared to calculate the values for the flow conductors. Both subroutines reference the flowrate array and an array containing conductor identification information. FLOCN1 assumes the specific heat is a function of temperature whereas, FLOCN2 assumes a constant value for specific heat.

2.1.3 Heat Exchanger Analysis

Four subroutines have been written to facilitate the thermal analysis of systems containing heat exchangers. These are HXCNT for analysis of counterflow heat exchangers, HXPAR for parallel flow heat exchangers, HXCROS for cross flow heat exchangers and HXEFF for any heat exchanger with an input effectiveness. These subroutines calculate the outlet temperatures of two sides based upon the inlet temperatures and heat exchanger effectiveness. The relations used for calculating effectiveness are described below.

2.1.3.1 Counterflow Heat Exchanger

Subroutine HXCNT calculates the heat exchanger effectiveness using the relation from Reference 3 for counterflow heat exchangers. That is,

\[
\epsilon = \frac{1 - e^{-\frac{UA}{(MC)_s} \left\{ 1 - \frac{(MC)_s}{(MC)_l} \right\} \left( MC \right)_s}}{1 - e^{-\frac{UA}{(MC)_s} \left\{ 1 - \frac{(MC)_s}{(MC)_l} \right\} \left( MC \right)_s}}
\]

Where

- \( \epsilon \) = effectiveness
- \( UA \) = overall effectiveness
- \( (MC)_s \) = mass, specific heat product for the side with the smallest MC
- \( (MC)_l \) = mass, specific heat product for the side with the largest MC

The limiting cases for this relation are:

(1) When \( (MC)_s / (MC)_l = 0 \),

\[
\epsilon = 1 - e^{-\frac{UA}{(MC)_s}}
\]

(2) When \( (MC)_s / (MC)_l = 1 \)

\[
\epsilon = \frac{UA}{1 + \frac{UA}{(MC)_s}} = \frac{UA}{(MC)_s + UA}
\]
Using the effectiveness as calculated by the above method, the outlet temperatures are calculated as follows:

1. For the side with the smallest MC, \((MC)_s\):
\[
T_{out_s} = T_{in_s} - \epsilon (T_{in_s} - T_{in_l}) \tag{9}
\]

2. The outlet temperature for the side with the large MC is then calculated by
\[
T_{out_l} = \frac{(MC)_s}{(MC)_l} (T_{in_s} - T_{out_s}) + T_{in_l} \tag{10}
\]

2.1.3.2 Parallel Flow Heat Exchanger

Subroutine HXPAR calculates the heat exchanger effectiveness using the relation for parallel flow heat exchangers, which is:
\[
\epsilon = 1 - e^{-\frac{UA}{(MC)_s \left[ \frac{1+(MC)_s}{(MC)_l} \right]}} \tag{11}
\]

The limiting cases are

(1) When \((MC)_s/(MC)_l = 0\),
\[
\epsilon = 1 - e^{-\frac{UA}{(MC)_s}}
\]

(2) When \((MC)_s/(MC)_l = 1\),
\[
\epsilon = \frac{-2 \frac{UA}{(MC)_s}}{2.0}
\]

The heat exchanger outlet temperatures are then calculated using equations 9 and 10.

2.1.3.3 Cross Flow Heat Exchanger

Subroutine HXCROS calculates the effectiveness for cross flow heat exchangers using one of the four relations below depending upon mixing of the streams.
Both Streams Unmixed

\[
\epsilon = 1 - e^{-\left(\frac{UA}{(MC)_s} \eta\right)} \left(\frac{(MC)_s}{UA}\right) \frac{1}{\eta}
\]

(12)

Where \( \eta = \left[\frac{(MC)_s}{UA}\right] 0.22 \)

Both Streams Mixed

\[
\epsilon = \frac{UA}{(MC)_s} - \frac{UA}{1 - e} \left(\frac{(MC)_s}{MC}_s\right) + \frac{UA}{1 - e} \left(\frac{(MC)_l}{MC}_l\right)
\]

(13)

Stream \((MC)_s\) Unmixed

\[
\epsilon = 1 - e^{-\left(\frac{(MC)_s}{MC}_s\right)} \left[1 - \frac{UA}{(MC)_s}\right]
\]

(14)

Stream \((MC)_l\) Unmixed

\[
\epsilon = 1 - e^{-\left(\frac{(MC)_l}{MC}_l\right)} \left[1 - \frac{UA}{(MC)_l}\right]
\]

(15)

The heat exchanger outlet temperatures are calculated using equations (9) and (10).

2.1.3.4 User Supplied Effectiveness

Subroutine HXEFF was written to perform heat exchanger thermal analysis with a user supplied effectiveness. The effectiveness may either be supplied as a constant or as an array number which gives the effectiveness as a bivariant function of the flowrates on the two sides. The outlet temperatures
are then calculated using equations (9) and (10).

2.1.4 Inline Heater Analysis

Provisions for the analysis of a fluid heater have been included in SINDA with subroutine HEATER. This subroutine simulates an electrical heater with a control system which turns the heater on when a specified sensor lump drops below a set value and turns the heater off when the specified sensor lump rises above another set value. When the heater is on an input quantity of heat is added to the heater node.

2.1.5 Cabin Analysis

A subroutine has been written for use with SINDA which will give the user the ability to perform thermal analyses on cabin air systems including condensation on the walls and a vapor mass balance. The cabin heat transfer and condensation analysis involves the two-component flow of a condensible vapor and a non-condensable gas, with condensation of the vapor occurring on surfaces in contact with the fluid. Two problems of this nature have been studied extensively.

1. Condensation on, or evaporation from, a surface over which a free stream of fluid is passing. In this case, for relatively low mass transfer rates, the fluid properties can be assumed to be constant.

2. Dehumidification of a confined fluid stream by a bank of tubes. In this case there is a marked change in the temperature and vapor content of the fluid, and the detailed deposition of the condensate is not of primary interest. This type of analysis is usually handled on an overall basis similar to heat exchanges effectiveness calculations.

The following additional assumptions have been made with respect to the cabin atmospheric conditions.

1. The heat of circulation in the cabin is sufficiently high that the temperature and humidity are effectively the same throughout the cabin.

2. The velocity at all points where heat transfer and/or condensation can occur is known, and is proportional to the total mass flow rate in the cabin.
These assumptions make it possible to calculate the heat and vapor balance in the cabin for the entire volume as a unit, and to solve the heat transfer and condensation equations at each node independently of the other nodes.

Cabin humidity can be determined from an overall vapor balance in the cabin. The total vapor in the cabin at the end of an iteration is:

\[ W_V = W_{V_i}^{-1} + W_{V_{in}} - W_{V_{out}} - \sum W_L \]

Where
- \( W_{V_i}^{-1} \) = mass of vapor in cabin at end of iteration \( i \)
- \( W_{V_i}^{-1} \) = mass of vapor in cabin at start of iteration \( i-1 \)
- \( W_{V_{in}} \) = mass of vapor flowing into cabin during iteration \( i \)
- \( W_{V_{out}} \) = mass of vapor flowing out of cabin during iteration \( i \)
- \( \sum W_L \) = mass of vapor condensed during iteration \( i-1 \)

\( W_{V_{in}} \) is determined from the known conditions of the gas flowing into the cabin.

\[ W_{V_{in}} = \dot{m}_{in} \left[ \frac{\psi_{in}}{1 + \frac{\psi_{in}}{\psi_{in}}} \right] \]

Where
- \( \dot{m}_{in} \) = mass flow rate into cabin
- \( \psi_{in} \) = specific humidity of gas flowing into cabin
- \( \Delta t \) = time increment

It is assumed that an equal volume of gas is flowing out of the cabin. Then,

\[ W_{V_{out}} = \dot{m}_{out} \left[ \frac{\psi_{c}}{1 + \frac{\psi_{c}}{\rho_{in}}} \right] \]

Where
- \( \psi_{c} \) = specific humidity in the cabin (at the end of the previous iteration)
- \( \rho_{c} \) = cabin density
- \( \rho_{in} \) = density of gas flowing into cabin

The condensation term \( \sum W_L \) is determined from the calculations for the individual nodes as described below. The properties of the cabin atmosphere are determined from the calculated value of \( W_V \). The vapor pressure
in the cabin is

\[ P_v = \frac{W_v}{V_c} R_v T_c \]

Where

- \( V_c \) = cabin volume
- \( R_v \) = gas constant
- \( T_c \) = temperature of cabin gas
- \( P_v \) = vapor pressure

Assuming that the cabin pressure \( P_c \) is a constant, the gas partial pressure \( P_a \) is:

\[ P_a = P_c - P_v \]

and

\[ W_a = \frac{P_a}{R_a T_c} \]

Where \( W_a \) = mass of non-condensible gas in the cabin.

Now the new value of specific humidity in the cabin can be determined by

\[ \psi_c = \frac{W_v}{W_a} \]

The properties of the atmosphere can now be determined by

\[ \mu_c = \frac{X \mu_g + \psi_c \mu_v}{X + \psi_c} \]

\[ C_{pc} = \frac{C_{pg} + \psi_c C_{pv}}{1 + \psi_c} \]

\[ k_c = \frac{X k_g + \psi_c k_v}{X + \psi_c} \]

\[ \rho_c = \frac{W_v + W_s}{V_c} \]

Where

- \( \mu \) = viscosity
- \( C_p \) = specific heat
- \( k \) = thermal conductivity
- \( X \) = molecular weight ratio, \( \frac{M_v}{M_g} \)
and all values are evaluated at \( T_c^{i-1} \). Cabin temperature \( T_c \) can be determined by a heat balance on the cabin atmosphere.

\[
T_c = T_c^{i-1} + \frac{\dot{m} \Delta T \Delta \bar{c}}{(W_v + W_A) C_p}
\]

Where \( T_c^{i-1} = \) \( T_c \) after previous iteration
\( \dot{m} \) = temperature of gas flowing into cabin
\( \Delta \bar{c} \) = net heat loss to cabin lumps

The heat transfer between the cabin atmosphere and the tube and structure lumps in the cabin is defined by:

\[
Q_{Li} = h_A \Delta T \Delta r
\]

Where \( h_A \) = heat transfer coefficient
\( \Delta T \) = heat transfer area of lump
\( \Delta r \) = time increment

Using the Colburn-Chilton heat transfer-mass transfer analogy, the condensation (or evaporation) at the tube lump is determined by:

\[
\Delta W_L = K_m A_L [P_v - P_{wi}] \Delta r
\]

Where \( W_L \) = condensation on wall, lb.
\( K_m \) = mass transfer coefficient
\( P_{wi} \) = vapor pressure at \( T_{Li} \)

The latent heat addition to the lump due to this condensation is

\[
\Delta Q_L = \Delta W_L \lambda
\]

Where \( \lambda \) = latent heat of vaporization

The vapor pressure \( P_{wi} \) can be determined by a relationship derived from the Clausius-Clapeyron equation and the perfect gas law (Appendix K of Reference 4).
The second option assumes flat plate flow for cabin wall lumps. In this case the heat transfer coefficient, for laminar flow, varies along the plate. Hence, direction of gas flow and the location of an assumed leading edge must be assumed. The equation for flat plates from Reference 3 is:

\[ N_u = 0.332 \cdot \left( Re \right)^{0.5} \cdot \left( Pr \right)^{1/3} \]

where the Nusselt and Reynold's numbers are local values and are defined by the distance X from the assumed leading edge. For a wall lump of length \( L_1 \) which is located a distance \( L_{i0} \) from the assumed leading edge, the
average Nusselt number can be defined as:

\[ \text{Nu} = 0.664 \Pr^{1/3} \left[ (Re)_{1}^{0.5} - (Re)_{0}^{0.5} \right] \]

Where \( \text{Nu} \) is defined by \( L_i \)
\( \text{Re}_0 \) is defined by \( L_{10} \)
\( \text{Re}_1 \) is defined by \( L_{10} + L_i \)

The third option is a direct user input for convective heat transfer coefficient.

For the determination of mass transfer coefficients, the same equations which were used for heat transfer coefficient can be used with the Sherwood number substituted for Nusselt number and Schmidt number for Prandtl number. However, if the diffusion coefficient for the cabin is approximately equal to thermal diffusivity, the Sherwood number is equal to the Nusselt number and the mass transfer coefficient can be determined directly from the heat transfer coefficient. That is:

\[ \text{Sh} = \text{Nu} \]

\[ \frac{K_m RT_{gX}}{D} = \frac{h_x}{k} \]

If \( D \approx \alpha \) then

\[ K_m = \frac{hD}{\alpha \rho C_p RT_g} \] (16)

\[ K_m \approx \frac{h}{C_p P_c} \]

Equation (16) is the Lewis relationship (Reference 3). For a mixture of oxygen and water vapor characteristic values are .866 for the diffusion coefficient, \( D \), and .879 for thermal diffusivity, \( \alpha \), so the relationship should be valid.

For cabin tube and wall lumps the values for \( \Delta Q_{L_i} \) and \( \Delta Q_{X_i} \) are added to the basic heat balance equation for these lumps. Values for \( \Delta Q_{L_i} \)
are summed for all participating lumps for input to the cabin atmosphere heat balance. Values for \( \Delta WL_i \) are also summed for all lumps for cabin humidity balance, and the value for total water condensed on each lump \( WL_i \) is maintained.

If the rate of evaporation or condensation is high it would be possible for the cabin humidity to change significantly during a single iteration. This could lead, for example, to overestimating condensation by assuming that the humidity is constant in the calculation. A test of the approximate vapor pressure in the cabin at the end of the iteration is made, and the condensation or evaporation at any lump is reduced, if the sign of the \( \Delta WL_i \) term is changed. A value \( W_v' \) is calculated by:

\[
W_v' = W_v^{i-1} - \Sigma WL_i
\]

and

\[
P_v' = \frac{W_v'}{144 \ V_c} R_v T_g
\]

Then for each lump if

\[
\frac{P_v' - P_{wi}}{P_v - P_{wi}} < 0
\]

a new value of \( \Delta WL_i \) is calculated by:

\[
\Delta WL_i = \Delta WL_i \left[ \frac{P_v - P_{wi}}{P_v' - P_{wi}} \right]
\]

The new values of \( \Delta WL_i \) are now again summed for the new value of \( \Sigma \Delta WL \) for establishing cabin humidity for the next iteration. A test is also made to assure that \( W_v' \) is never less than zero.

2.1.6 Radiation Interchange Analysis

Capabilities have been incorporated into subroutines for use with SINDA to facilitate the analysis of radiation heat transfer in an enclosure. The capabilities include the ability to:
(1) Analyze diffuse and/or specular infrared radiation in an enclosure
(2) Analyze diffuse and/or specular radiation from an external source for as many wave bands as desired
(3) Consolidate several temperature nodes into a single surface to improve computational efficiency

A radiation surface is defined as a group of temperature nodes which may be assumed to have identical radiating properties, angle factors and interchange factors.

The subroutines account for the net radiation heat transfer between a number of surfaces due to emitted radiation from each surface, reflected radiation from each surface, and radiation from any number of incident sources. The reflection of the energy originally emitted by another surface or from an external source may be either diffuse, specular, or any combination of the two.

2.1.6.1 Emitted Radiation In A Cavity

The radiosity of a surface is defined as the flux of infrared radiation leaving that surface with a diffuse distribution (according to Lambert's Law). That energy leaving a surface which has been reflected in a specular manner does not contribute to the radiosity of that surface. The incident infrared radiosity is denoted by the symbol H. The reflectance \((1 - \varepsilon)\) of a surface is separated into two components, the diffuse reflectance \((\rho)\) and the specular reflectance \((\rho^S)\). Here \(\varepsilon\) is the emittance of the surface and is equivalent to the absorptance for long wavelength radiation. With the angle factors \((F_{ij})\) defined in the normal way, there exist similar angle factors which relate the geometrical ability of surface \(i\) to radiate to surface \(j\) by means of a mirror-like reflection from specular surface \(k\). Reference to Figure 1 indicates the method of imagery which will enable the calculation of these reflected angle factors. Here the angle factor to surface \(j\) is identical with the angle factor to the image of surface \(j\). Also the angle factor is limited by the ability of surface \(i\) to "see" through the "window" of surface \(k\).

With the specular surface angle factors so defined, an interchange factor \(E_{ij}\) is defined similarly to reference 5 as follows:

\[
E_{ij} = \sum_k \rho^S_k F_{ij}(k) + \sum_k \sum_l (\rho^S_k \rho^S_l) F_{ij}(k,l) + \ldots
\]
Figure 1: Illustration of method used to determine specular surface reflected view factors.
Here $F_{ij}(k)$ is the angle factor from $i$ to $j$ as seen in the specular surface $k$, $F_{ij}(k,1)$ is the angle factor from $i$ to $j$ as seen in the double specular reflection from $k$ and $1$. There are an infinite number of possible combinations of these multi-reflections. It is evident that the interchange factors account for the specularly reflected radiant flux from the reflecting surface. This portion of total leaving flux is not a component of the radiosity of that surface. The radiosity may be written

$$B_i = \epsilon_i \sigma T_i^4 + \rho_i H_i,$$

and, for $ns$ surfaces,

$$H_i = \frac{1}{A_i} \sum_{j=1}^{ns} B_j A_j E_{ij}$$

Now the interchange factors obey the reciprocity relation

$$A_i E_{ij} = A_j E_{ji}$$

So,

$$H_i = \sum_j B_j E_{ij}$$

Substitution into the equation for $B$ results in

$$\sum_j (\delta_{ij} - \rho_i E_{ij}) B_j = \epsilon_i \sigma T_i^4$$

This equation represents a set of linear, simultaneous, inhomogeneous algebraic equations for the unknowns ($B_j$). The symbol $\delta_{ij}$ is the Kronecker delta function which is 1 when $i = j$ and is 0 when $i \neq j$.

Note that the coefficients of $B_j$ in equation (22) do not form a symmetric coefficient matrix since the off diagonal terms contain $-\rho_i E_{ij}$. This equation can be made symmetric by multiplying each equation by $A_j/\rho_j$. 

17
This gives
\[ \sum_j \left( \frac{B_{ij} A_i}{\rho_i} - E_{ij} A_i \right) B_j = \frac{\epsilon_i A_i}{\rho_i} \sigma T_i^4 \quad i = 1, \ldots, ns \] (23)

Written in matrix form this equation is
\[ EB = T \] (24)

Where \( E \) is a symmetric coefficient matrix. The solution is
\[ B = E^{-1} T = [e^{-1}] T \] (25)

or
\[ B_i = \sum_{j=1}^{ns} e_{ij}^{-1} \frac{\epsilon_i A_i}{\rho_j} \sigma T_j^4 \] (26)

The net heat transfer rate absorbed by surface \( i \) is given by
\[ Q_i = A_i \epsilon_i \left[ H_i - \sigma T_i^4 \right] \] (27)

Where \( H_i \) is given from equation (18) as
\[ H_i = \frac{1}{\rho_i} \left[ B_i - \epsilon \sigma T_i^4 \right] \]

Substituting in for \( H_i \) gives
\[ Q_i = A_i \epsilon_i \left\{ \frac{1}{\rho_i} \left[ B_i - \epsilon \sigma T_i^4 \right] - \sigma T_i^4 \right\} \]
\[ = \frac{A_i \epsilon_i}{\rho_i} \left\{ B_i - [\rho_i + \epsilon_i] \sigma T_i^4 \right\} \] (28)
Substituting in for $B_i$ from equation (26) into equation (28) gives

$$Q_i = \frac{A_i \epsilon_i}{\rho_i} \left\{ \sum_{j=1}^{n_s} \left( \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_j} \sigma T_j^4 - \frac{\rho_i + \epsilon_i}{\rho_i} \sigma T_i^4 \right) \right\}$$

$$= \frac{A_i \epsilon_i}{\rho_i} \left\{ \sum_{j=1, j \neq i}^{n_s} \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_j} \sigma T_j^4 - \left[ \rho_i + \epsilon_i - \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_i} \right] \sigma T_i^4 \right\} \quad (29)$$

Since, in steady state, $Q_i = 0$, and $T_i^4 = T_j^4$ for all $i$ and $j$ we can conclude that

$$\rho_i + \epsilon_i - \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_i} = \sum_{J=1, J \neq i}^{n_s} \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_j} \quad (30)$$

Making the above substitution in equation (29) gives

$$Q_i = \sum_{J=1}^{n_s} \sigma \frac{\epsilon_i \epsilon_j A_j A_j e_{ij}^{-1}}{\rho_i \rho_j} \left[ T_j^4 - T_i^4 \right] \quad (30)$$

If we define $F$ as

$$F_{ij} = \frac{\epsilon_i \epsilon_j A_j e_{ij}^{-1}}{\rho_i \rho_j} \quad (31)$$

$$F_{ij} = \frac{\epsilon_i \epsilon_j A_j}{\rho_i \rho_j} \left[ e_{ij}^{-1} - \frac{\rho_i}{A_j} \right] \quad i = j$$

Then

$$Q_i = \sum_{J=1}^{n_s} \sigma F_{ij} A_i \left[ T_j^4 - T_i^4 \right] \quad (32)$$

This equation gives the heat flux between surfaces. However, each surface can contain several nodes. The heat absorbed by for each node is determined by:
\[ Q_n = \frac{A_n}{A_i} \sum_{j=1}^{n_s} \sigma \Phi_{ij} A_i \left[ T_j^4 - T_i^4 \right] \]  

(33)

Where \( n \) = the node number on surface \( i \)

Prior to each iteration, the temperature of the surfaces are determined by

\[ T_i^4 = \frac{\sum_{n=1}^{n_n} A_n T_n^4}{\sum_{n=1}^{n_n} A_n} \]

(34)

Where \( n_n \) = the number of nodes on surface \( i \)

Since the heat transfer rate given by equation (33) depends on the node temperature, stability considerations must be taken into account. This is handled by storing the following relation into the array containing the sum of the conductors used for time increment calculation

\[ \text{CON}_n = 4 \frac{A_n}{A_i} \sigma T_n^3 \sum_{j=1}^{n_c} \Phi A_{ij} \]  

(35)

Subroutine RADIR makes the calculations necessary to obtain \( Q_n \) given by equation (33) and \( \text{CON}_n \) given by equation (35). The following is a summary of the calculations:

A. The following are performed the first time through RADIR:
   1. From the user input values of \( E_{ij}, A_i \), and \( p_i \), the \( E \) matrix given by equation (24) is formed. Only half of the symmetric matrix is stored to save space.
   2. The \( E \) matrix is inverted in its own space to get \( E^{-1} \) with elements \( e_{ij}^{-1} \)
   3. The \( \Phi A_{ij} \) values are determined from equation (31) and stored in the surface connections data.

B. The following calculations are performed on each temperature iterations:
   1. The temperature of each surface is calculated by equation (34).
   2. The heat absorbed for each node is determined using equation (33) and is added to the \( Q \) array.
The routine utilizes data used for obtaining $F_{Aij}$ in step A as working space for step B, thus, maximizing space utilization.

2.1.6.2 Radiation From External Source

As with the internally generated radiation, the solar (or any other external source radiation) interchange factor is defined by

$$E_{ij}^* = F_{ij} + \sum_k \rho_k^* F_{ij}(k) + \sum_k \sum_l \rho_l^* \rho_k^* F_{ij}(k,l) + \ldots$$

Where $\rho_k^*$ is the solar specular reflectance of surface $K$

$F_{ij}(k)$ is the angle factor from $i$ to $j$ as seen in the specular surface $K$

$F_{ij}(K,l)$ is the angle factor from $i$ to $j$ as seen in a double specular reflection from $j$ to $l$ back to $i$

The interchange factors as defined above account for the specularly flux reflected from the surface. Thus, since the specular component of the flux is assumed to go directly from surface $i$ to surface $j$ by the interchange factor, $E_{ij}$, this portion of the total flux is not a component of the radiosity for the intermediate surfaces ($k$ and $l$ above). The radiosity of surface $i$ is given by

$$B_i^* = \rho_i^* H_i^*$$

Where $B_i^*$ is the radiosity (energy leaving)

$H_i^*$ is the incident energy

$\rho_i^*$ is the diffuse reflectance

The energy incident upon a surface is given by

$$H_i = \sum_{J=1}^{ns} B_j^* E_{ij}^* + S_i$$

Where $S_i$ is the energy directly incident on surface $i$ from an external source
Substituting equation (36) into (37), multiplying by $A_i/\rho_i^*$ and simplifying gives the following relation for the radiosity

\[
\begin{bmatrix}
A_i - E_{ij}^* A_i
\end{bmatrix}
B_i^* = \sum_{j=1, j \neq i}^n E_{ij}^* A_i B_j^* = S_i A_i \quad i=1,n
\]

This set of $n$ equations can be written in matrix form as

\[
E^* B^* = S
\]  

(38)

Note that the equations are written so that $E^*$ is a symmetric matrix, which has the solution for $B^*$

\[
B^* = E^{*-1} S \quad \text{or} \quad B_i = \sum_{j=1}^n [e_{ij}^*]^{-1} S_j A_j
\]  

(40)

Where $[e_{ij}^*]^{-1}$ is the $ij$th element of the inverse of the $E^*$ matrix.

The heat flux absorbed by the $i$th surface is given by

\[
\frac{Q_i^*}{A_i} = a H_i
\]  

(41)

But from equation (36)

\[
H_i = \frac{B_i}{\rho_i^*}
\]  

(42)

Combining equations (40), (41), and (42) gives

\[
Q_i^* = \sum_{j=1}^n e_{ij}^* S_j A_j A_i S_j
\]  

(43)

If we define

\[
\sigma_{ij}^* = e_{ij}^* \frac{a_i}{\rho_i^*} A_j
\]  

(44)

Then the absorbed heat flux is given by

\[
Q_i^* = \sum_{j=1}^n \sigma_{ij}^* A_i S_j
\]  

(45)
Equation (45) gives the heat absorbed by each surface. However, each surface may contain several temperature nodes. The absorbed heat for each node is given by:

\[ Q_n^* = \frac{A_n}{A_i} Q_i^* \]  

(46)

Where \( A_n \) is the area of the node

Subroutine RADSOL was written to make necessary calculations to obtain \( Q_n^* \) given by equation (46). The following is a summary of the calculations:

A. The following calculations are made the first time through RADSOL:

1. From the user input values of \( E_{ij}^*, \rho_i^* \), and \( A_i \), the \( E^* \) matrix given by equation (39) is formed. Only one half is stored since \( E^* \) is symmetric.
2. The \( E^* \) matrix is inverted in its own space to get \( E^{-1} \) with elements, \( e_{ij}^{-1} \).
3. The \( \mathcal{F}_{ij}^* A_i \) values are determined from equation (44) and stored in the surface connections data.

B. The following calculations are performed on each temperature iteration:

1. The heat flux absorbed by each node is calculated by

\[ \frac{Q_i^*}{A_i} = \frac{1}{A_i} \sum_{j=1}^{n} \mathcal{F}_{ij}^* A_j S_j \]

2. The net heat absorbed by this wavelength radiation is calculated for each temperature node on each surface by

\[ Q_n^* = A_n \frac{Q_i^*}{A_i} \]

This quantity of absorbed heat is added to the \( Q \) array for node \( n \).
Note that the user may specify subroutine RADSOL for as many bands of radiation from an external source as desired. A single call is required for each band.
2.2 Fluid Flow Analysis

Subroutine PFCS was written as a SINDA user subroutine to provide the ability to perform fluid pressure/flow analysis for flow of an incompressible fluid in tubes. The fluid flow analysis of PFCS is integrated with the thermal analysis capability so that the temperature dependence of properties is included in the pressure balances. PFCS is called from the VARIABLES 2 user logic block.

PFCS performs a pressure-flow balance on a general flow network including the following effects:

1. Friction pressure drop
2. Orifices and fitting type pressure losses
3. Valves
4. Pumps
5. Incoming flow sources at any pressure point in the system

The user describes the flow model to the subroutine by supplying the tube network connections and information concerning fluid properties, flow geometry, temperature model lumps, orifices, valves and pumps. Using this information, the subroutine determines the flow distribution required to satisfy (1) the conservation of mass at each node point and (2) equal pressure drops across tubes in parallel. The model used to describe the flow system and the analytical methods for determining the solution are described below.

2.2.1 Overall Flow Model Description

A flow problem may be analyzed with PFCS, simultaneously with a thermal analysis, so that the flow solution is continually updated based on the thermal conditions. To perform a flow analysis, the user must input a mathematical model of the flow system. The flow system is assumed to consist of a set of interconnected tubes such as the example shown in Figure 2 which consists of two radiator panels, each containing four tubes and connected so that they flow in parallel.

For clarity the following definitions are made at this point:
FIGURE 2 FLOW SYSTEM SCHEMATIC

Tube Numbers

Pressure Nodes

FIGURE 2 FLOW SYSTEM SCHEMATIC
(1) A tube is any single length of pipe between two pressure nodes. A tube "contains" fluid temperature nodes and may contain as many of these as required.

(2) A pressure node is located at each end of a tube. As many tubes as desired may be connected at a node junction and a node must exist at the junctions of two flow pipes.

We must make a mathematical model to describe the fluid flow information to the computer. The information required consists of:

(1) Identification of the pressure node numbers
(2) Identification of the tube numbers and the two pressure nodes connected by tube
(3) The fluid temperature nodes contained in each tube
(4) The flow geometry for each temperature fluid nodes
(5) The number of "head losses" for items such as orifices
(6) Fluid property information
(7) Valve connections and characteristics
(8) Pump characteristics

To build a flow mathematical model, a schematic of the flow system is needed. As shown in Figure 2, the pressure nodes and tubes may be superimposed on the schematic. It is also helpful to impose the fluid temperature lump numbers for each tube.

To facilitate speedy analysis on a general flow problem, provisions have been made for the user to divide the flow system network into subnetwork elements. For example, the flow system shown in Figure 2 could be divided as shown in Figure 3. Tubes 23 and 24 are added in the main network as shown in 3(a) to replace subnetwork elements 1 and 2. The subnetwork elements 1 and 2 which are shown in Figures 3(b) and 3(c) are then input as separate network elements. This type of subdivision allows the solution to be obtained by solving two sets of 6 simultaneous equations and one set of 8 equations rather than the original set of 16 simultaneous equations. This type of subdivision has been found to enhance the solution speed and accuracy for problems with a large number of nodes.

In summary, the pressure/flow solution is obtained by the following sequence:
a) Main Flow Network

b) Subnetwork No. 1

c) Subnetwork No. 2

FIGURE 3 MAIN NETWORK AND SUBNETWORKS
(1) The flow resistance is obtained for each fluid temperature lump in each tube including the effects of friction, orifices, and fitting type losses.

(2) The flow conductor valve is obtained for each tube by summing all the resistances of the fluid lumps in the tube, adding the valve and user supplied resistance to the sum, and inverting the resistance.

(3) A set of simultaneous equations is set-up and solved for each main system and subnetwork to obtain the pressures.

(4) The flow rates are then calculated.

A detail discussion of each element in the above sequence is described in the following subsections.

2.2.2 Tube Conductor Determination

The value of the flow conductor is determined for each tube by first calculating the flow resistance for each temperature fluid lump contained in the tube, summing these resistances up to obtain the flow resistance of the tube and inverting the tube resistance to get the conductance.

Flow conductance is defined by the relationship

\[ \dot{W}_{ij} = GF_{ij} \{P_i - P_j\} \] \hspace{1cm} (47)

Where

- \( \dot{W}_{ij} \) = flow rate between pressure nodes \( i \) and \( j \)
- \( GF_{ij} \) = flow conductance between nodes \( i \) and \( j \)
- \( P_i \) = pressure at pressure node \( i \)
- \( P_j \) = pressure at pressure node \( j \)

The flow resistance for each lump is then

\[ R_k = \frac{1}{GF} = \frac{\Delta P_k}{\dot{W}} \]

Where

- \( R_k \) = flow resistance for lump \( k \)
- \( \Delta P_k \) = pressure drop for lump \( k \)

But \( \Delta P_k \) is given by

\[ \Delta P_k = \left( f_k \cdot ffc \cdot \frac{L_k}{D_k} + K \right) \frac{W^2}{2g_c \rho_k \kappa A^2} \] \hspace{1cm} (48)
Where \( f_k \) = the friction factor for lump \( k \)

\( \text{ffc} \) = the friction factor coefficient

\( L_k \) = the lump length for lump \( k \)

\( D \) = the lump hydraulic diameter for lump \( k \)

\( K \) = the dynamic head losses for lump \( k \)

\( \dot{W} \) = the flow rate

\( g_c \) = the gravitational constant

\( \rho_k \) = the fluid density for lump \( k \)

\( A \) = the flow area

The flow resistance is then given by

\[
R_k = \left( f_k \text{ffc} \frac{L_k}{D_k} + K \right) \frac{\dot{W}}{2g_c \rho_k A^2} 
\]

(49)

Two options are available for obtaining the friction factor, \( f_k \). These are (1) internal calculations for all flow regimes and (2) internal calculation for laminar flow and obtained from a table of \( f \) vs \( \text{Re} \) (where \( \text{Re} \) is the Reynold's number) for transition and turbulent flow. For the first option, the internal calculations for the three flow regimes are:

**Laminar Regime:** \( \text{Re}_k \leq 2000. \)

\[
f_k = \frac{64}{\text{Re}_k} \]

(50)

Where \( f_k \) = friction factor for lump \( k \)

\( \text{Re}_k \) = Reynolds number for lump \( k \)

**Transition Regime:** \( 2000 < \text{Re}_k < 4000 \)

\[
f_k = .2086082052 - .1868265324 \left[ \frac{\text{Re}_k}{1000} \right] \]

\[
+ .06236703785 \left[ \frac{\text{Re}_k}{1000} \right]^2 - .0065545818 \left[ \frac{\text{Re}_k}{1000} \right]^3
\]

(51)
**Turbulent Regime**: \( Re_k \geq 4000 \)

\[
f_k = \frac{316}{(Re_k)^{.25}}
\]

Equation (51) for the transition regime is a curve fit between the laminar and turbulent regimes which was derived to match the two curves in a continuous manner. It is merely an arbitrary curve in this undefined region. A curve of the friction factor vs Reynold's number given by the above relations is shown in Figure 4.

The second option for friction factor uses equation (50) for the laminar regime and a user input curve of \( f_k \) vs \( Re \) for the other regimes. The options available for input of the dynamic head loss, \( K \), include (1) an input constant or (2) a tabulated curve of \( K \) vs \( Re \).

To obtain the conductance for each tube, the flow resistances for all the lumps in the tube are added and then inverted, giving

\[
GF_{ij} = \frac{1}{\sum_k R_k}
\]

### 2.2.3 Valve Analysis

Provisions have been included in subroutine PFCS for valves to be included in the flow balance. The valve pressure drop is characterized by the following equation for each side of the valve;

\[
\Delta P = E \left[ \frac{\dot{\omega}}{X} \right]^2
\]

where
- \( \Delta P \) = valve pressure drop
- \( E \) = valve pressure drop factor (user input)
- \( \dot{\omega} \) = flowrate through the side of the valve under consideration
- \( X \) = the fraction of the valve opening (\( X = 0 \) indicates valve closed; \( X = 1.0 \) indicates valve full open)
FIGURE 4  Friction Factor vs Reynolds Number

Re \times 10^{-3} (curve 1)
Re \times 10^{-4} (curve 2)
Three basic types of valves are available in PFCS which give different characteristics for the dynamics of the valve position $X$. These types are:

1. Rate Limited
2. Polynomial
3. Shut-off

A number of variations are available for each valve type. For instance, each of the above may be either one sided or two sided. If a valve is two sided, the valve position of side 2, $X_2$, is related to that of side one by

$$X_2 = 1.0 - X_1$$

If the valve is one sided, either side one or side two may be used. Provisions are included for a valve time constant to be included with the polynomial valve.

The methods used to obtain the valve positions for each of the three methods are described below.

2.2.3.1 Rate Limited Valve

The valve position for the rate limited valve is obtained by an approximate integration of the valve rate of movement, $\dot{X}$. $\dot{X}$ depends on the temperature difference between the valve control set point temperature and the sensor temperature as shown in Figure 5. With this characteristic, the valve has no movement as long as the valve temperature error, $\Delta T$, is within the dead band. Outside the dead band, the velocity of the valve increases linearly as the error increases to a maximum rate, $\dot{X}_{\text{max}}$. The dead band, rate of velocity increase, $d\dot{X}/dT$, and the maximum velocity are controlled by user input.

The relations used to obtain the valve positions are as follows:

$$X_{i+1} = X_i + (\dot{X}_{i+1}) (\Delta r)$$

Where

- $X_{i+1}$ = valve position at iteration $i+1$
- $X_i$ = valve position at iteration $i$
- $\dot{X}_{i+1}$ = valve velocity at iteration $i+1$
- $\Delta r$ = the problem time increment
Valve Position, \( X = x_0 + \dot{X} \text{ (Time Increment)} \) \( 0 < x < 1.0 \)

\[ \text{Side 2 } \dot{X} \text{ Max} \]

\[ \frac{d\dot{X}}{d\Delta T} = \text{constant} \]

\[ \dot{X} \text{ Max } \text{Side 1} \]

\[ \dot{X}_{\text{min}} = -\dot{X}_{\text{max}} \]

\[ \Delta T = \text{Sensor Temp} - \text{Set Point} \]

Figure 5 Rate Limited Valve Operation
The valve position is limited by

\[ X_{\text{min}} \leq x^{i+1} \leq X_{\text{max}} \]

Where \( X_{\text{min}} \) and \( X_{\text{max}} \) are input limits on the valve position.

The valve velocity, \( \dot{x}^{i+1} \), in equation (54) is given by:

\[ \dot{x}^{i+1} = 0 \text{ if } |T_{\text{sen}} - T_{\text{set}}| \leq T_{\text{db}} \]

Where

- \( T_{\text{sen}} \) = Sensor lump temperature
- \( T_{\text{set}} \) = Set point temperature
- \( T_{\text{db}} \) = Valve dead band temperature

\[ \dot{x}^{i+1} = \frac{d\dot{x}}{d(\Delta T)} \begin{cases} [T_{\text{sen}} - T_{\text{set}} - T_{\text{db}}] & \text{if } T_{\text{sen}} > T_{\text{set}} + T_{\text{db}} \\ [T_{\text{sen}} - T_{\text{set}} + T_{\text{db}}] & \text{if } T_{\text{sen}} < T_{\text{set}} - T_{\text{db}} \end{cases} \]

The valve velocity is limited by

\[ \dot{x}_{\text{min}} \leq \dot{x}^{i+1} \leq \dot{x}_{\text{max}} \]

After the valve position for side 1 is obtained from equation (54), the side 2 position is obtained from \( x_2 = 1.0 - x_1 \)

2.2.3.2 Polynomial Valve

The polynomial valve determines the steady state valve position as a 4th degree polynomial function of the temperature error between the sensor lump and the set point. A valve time constant is then applied to determine how far between the previous position and the new steady state position the valve will move. The steady state position, \( X_{\text{ss}} \), is given by

\[ X_{\text{ss}} = A_0 + A_1 \Delta T + A_2 \Delta T^2 + A_3 \Delta T^3 + A_4 \Delta T^4 \]

Where \( \Delta T = T_{\text{sen}} - T_{\text{set}} \)

- \( T_{\text{sen}} \) = the sensor lump temperature
- \( T_{\text{set}} \) = the set point temperature
- \( A_0, A_1, A_2, A_3, A_4 \) = input constants
The valve position, $x^{i+1}$ is then determined by

$$x^{i+1} = x_{ss} + (x^i - x_{ss}) e^{-\Delta t/r_c}$$

Where

- $x^{i+1}$ = valve position at iteration $i+1$
- $x^i$ = valve position at iteration $i$
- $\Delta t$ = problem time increment
- $r_c$ = valve time constant

The valve position for side 2 is given by

$$x_2 = 1.0 - x_1$$

where $x_1$ is given by equation (55).

Note that this valve combines the capabilities of the polynomial valve and the proportioning valve described in Reference 6. If one desires to eliminate the effect of the time constant (and thus, give the valve an instantaneous response), a value for $r_c$ should be input which is small compared to the time increment, $\Delta t$. Also, either a constant value or a temperature lump number may be specified for the set point to permit use of the valve for proportioning between two sides.

2.2.3.3 Shut-off Valve

For side 1 of a shut-off valve the valve position decreases from $x_{\max}$ to $x_{\min}$ when the temperature of the sensor lump drops below the specified "off" temperature, $T_{off}$, and increased from $x_{\min}$ to $x_{\max}$ when the sensor lump exceeds a second specified temperature, $T_{on}$. $T_{on}$ must be greater than $T_{off}$. Side 2 works in reverse of side 1. The valve position increased from $x_{\min}$ to $x_{\max}$ when the sensor temperature drops below the specified $T_{on}$ and decreases from $x_{\max}$ to $x_{\min}$ when the sensor lump increases above the off temperature, $T_{off}$. For side 2, $T_{off}$ must be greater than $T_{on}$. Note that, if the shut-off valve is a two sided valve with both sides active, the valve is a switching valve.

2.2.3.4 Valve Flow Resistance Calculations

The valve pressure drop on side one is assumed to be given by:
\[ \Delta P = E \left( \frac{\dot{W}}{X} \right)^2 \]  \tag{56}

Where $E$ is an input constant

$W$ is the flow through one side of the valve

$X$ is the valve position (fraction of total possible distance)

Since flow resistance is $\Delta P/\dot{W}$, the valve flow resistance is given by

\[ R_v = \frac{E \cdot \dot{W}}{X^2} \]  \tag{57}

This value of flow resistance is calculated and added to the other flow resistances of the tube prior to performing the operation in equation (53) to find the value of the flow conductor for the tube.

Valves may be either one way or two way - i.e., be one tube or two tubes at the outlet. If only one tube exists on the valve outlet the flow resistance is calculated using equation (57) above. If a second tube exists, the resistance on side 2 is given by

\[ R_{v2} = \frac{E_2 \dot{W}_2}{(1 - X)^2} \]  \tag{58}

2.2.4 Pressure-Flow Network Solution

As previously stated, the user may subdivide a system flow network into a main network and subnetwork elements. The elements which are subnetworks to the main network may also contain subnetwork elements but the subdivision can go no lower than two levels.

After the flow conductor values have been obtained by the methods described in Sections 2.2.2 and 2.2.3 a set of simultaneous equations are set up and solved for the main system and for each subnetwork. The subnetwork elements are all solved first and then, their equivalent flow conductor value is calculated. The value is inserted in the main system network and the system solution is obtained. The procedure is repeated until the problem is balanced.

A set of simultaneous equations are obtained by conservation of mass at each pressure node for each network and subnetwork. For any node $i$ the conservation equation can be written as follows:
\[ \sum W_{\text{out}} - \sum W_{\text{in}} = 0 \quad (59) \]

Let \[ W_{\text{in}} = W_i \]
and \[ \sum W_{\text{out}} = \sum_{j=1}^{\text{nc}} GF_{ij} [P_j - P_i] \]

Then equation (59) becomes
\[ \sum_{j=1}^{n} GF_{ij} [P_j - P_i] - W_i = 0 \quad i=1,n \quad (60) \]

Where \[ GF_{ij} \] = flow conductor between pressure nodes \( i \) and \( j \)
\[ P_i = \text{pressure at node} \ i \]
\[ P_j = \text{pressure at node} \ j \]
\[ W_i = \text{flow rate added at node} \ i \]
\[ n = \text{number of pressure nodes in the subnetwork} \]

The above equation is a set of \( n \) simultaneous equations for \( P \) array. Pressure in the system or subsystem may be set at a specified level but the last (outlet) node must be specified. Equation (60) may be written in matrix form as:
\[ GP = C \]

Where
\[
G = \begin{bmatrix}
\sum GF_{ij} - GF_{i2} - GF_{i3} & \cdots \\
-GF_{21} & \sum GF_{2j} - GF_{23} & \cdots \\
\vdots & \vdots & \ddots \\
-GF_{n-1,1} & -GF_{n-1,2} - \cdots - GF_{n-1,j} & \cdots
\end{bmatrix}
\]
\[ P = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ \vdots \\ P_{n-1} \end{bmatrix} \]

\[ C = \begin{bmatrix} W_1 + GF_{1n} P_n \\ W_2 + GF_{2n} P_n \\ \vdots \\ \vdots \\ W_{n-1} + GF_{n-1n} P_n \end{bmatrix} \]

\( P_n \) is the specified pressure. The above equations are solved for pressures at each point in the system and flow rates are then calculated for each tube by:

\[ \dot{W}_{ij} = GF_{ij} (P_i - P_j) \quad (62) \]

Since the coefficient matrix given by equation (61) is symmetric and positive definite the efficient square root or Symmetric Cholesky method was programmed to obtain the solution. This method is more accurate and faster than any other methods studied for this application.

Since the flow conductors are functions of the flow rate, the set of equations given by (61) are solved numerous times on each temperature iteration with a net set of \( GF_{ij} \) values for each solution. The iteration
process continues until the change in the flowrates is within some user specified tolerance before proceeding to the next iteration.

2.2.5  Pump and System Pressure - Flow Matching

Concurrent with iterating the system flow equation to solution on each temperature iteration, the overall system pressure drop and flowrate must be matched to a pump characteristic. Several types of pump characteristics are available to the user as options. These are (1) system flow rate specified as a constant, (2) system flowrate specified as a known function of time, (3) pressure drop specified as a function of the flowrate in a tabulated form and (4) pressure drop specified as a function of flowrate with a fourth degree polynomial curve.

The first two options require no balancing of the pump with the system. Balancing is required for options (3) and (4) and iterative procedures have been devised to obtain the solution of the pump curve to the system characteristics with as few passes as possible through the system pressure/flow balancing loop for these options. The procedures used for these options are described below.

2.2.5.1 Tabulated Pump Curve Solution

The matching of a tabulated pump pressure rise/flow characteristic to the system pressure drop/flow characteristic is accomplished by the following procedure. See Figure 6 to aid in understanding the procedure.

Step 1: The initial flowrate, \( W_1 \), at the system inlet is established either from user input on the first iteration or the system flow of the previous iteration for subsequent iterations.

Step 2: Using \( \dot{W}_1 \) a solution to the flow network is obtained using the methods described in Sections 2.2.2, 2.2.3 and 2.2.4. Following this solution, \( \Delta P_1 \) is available establishing point 1 on the true system characteristic curve shown in Figure 6.

Step 3: Obtain an equation for the straight line approximation of the system characteristic (line 0, 1 for the first pass, line 1, 2 for the second pass, etc.)

\[ \Delta P_S = C W_S + D \]
FIGURE 6 SYSTEM/PUMP CURVE SOLUTION
where \( C = \frac{\Delta P_1 - \Delta P_0}{W_1 - W_0} \)

\[
D = \Delta P_0 - \frac{\Delta P_1 - \Delta P_0}{W_1 - W_0} W_0
\]

\( \Delta P_s, W_s \) are the system pressure drop and flowrate values given by the approximate equation

\( \Delta P_1, W_1 \) are the latest values for system pressure drop and corresponding system flowrate

\( \Delta P_0, W_0 \) are the values for system pressure drop and corresponding system flowrate for the previous pass (These values are zero for the first pass)

Step 4: Obtain the equation of the line connecting points \( a_1 \) and \( b_1 \) which is an approximation of the pump characteristic.

(1) Two points are determined on the pump characteristic curve:
   (a) interpolate the tabulated characteristic at \( W_1 \) to obtain \( \Delta P_{a1} \) (See Figure 6) to locate point \( a_1 \) at \( W_1, \Delta P_{a1} \). If \( W_1 \) is greater than \( W_{\text{max}} \), set \( W_1 \) equal to \( W_{\text{max}} \) and \( \Delta P_{a1} \) equal to zero.
   (b) reverse interpolate the tabulated characteristic at \( \Delta P_1 \) to obtain \( W_{b1} \) to locate point \( b_1 \) on the curve. If \( \Delta P_1 \) is greater than \( \Delta P_{\text{max}} \), \( \Delta P_1 \) is set to \( \Delta P_{\text{max}} \) and \( W_{b1} \) is set to zero.

(2) Determine the coefficients \( A \) and \( B \) for the equation

\[
\Delta P_p = AW_p + B
\]

where \( A = \frac{\Delta P_1 - \Delta P_{a1}}{W_{b1} - W_1} \)

\[
B = \Delta P_{a1} - \frac{\Delta P_1 - \Delta P_{a1}}{W_{b1} - W_1}
\]
\( \Delta P_p, W_p \) are the pump pressure rise and flowrate as given by the approximation.

Step 5: Solve the approximate equations obtained in Steps 3 and 4 to obtain an approximate solution to the system characteristic and the pump characteristic (Point N) as follows:

\[
W_N = \frac{D - B}{A - C}
\]

\( \Delta P_N = AW_3 + B \)

Step 6: Check the tolerance below where \( W_{N-1} \) is the previous \( W_N \) (\( W_1 \) for the first time through)

\[
\text{Is} \quad \frac{W_N - W_{N-1}}{W_{N-1}} < .001
\]

(1) If the above inequality equation is not satisfied repeat steps 4 through 6 substituting \( W_N \) for \( W_1 \) and \( \Delta P_N \) for \( \Delta P_1 \)
(2) If the inequality is satisfied the point \( S_1 \) (Figure 6) has been located. Continue with step 7. The final flowrate is \( W_2 \)

Step 7: Check the following tolerance

\[
\text{Is} \quad \frac{W_2 - W_1}{W_1} < \text{TOL}^*
\]

(1) If the above inequality equation is not satisfied, repeat steps 2 through 7 using the value of \( W_2 \) for \( W_1 \).
(2) If the inequality is satisfied, \( W_2 \) is the solution flowrate.

*\text{TOL} is the input pressure solution tolerance described on page 74
2.2.5.2 Polynomial Pump Curve Solution

When the user describes the pump curve with a polynomial curve fit, the pump characteristic is described by the relation

\[ \Delta P_p = A_0 + A_1 W + A_2 W^2 + A_3 W^3 + A_4 W^4 \]

When this option is used, the procedure for matching the pump characteristic to the system characteristic is identical to that described in Section 2.2.5.1 for the tabulated pump characteristic except Steps 4 and 5 are replaced with the following:

Step 4: Obtain the coefficients of the 4th order equation to be solved
Since:

\[ \Delta P_p - \Delta P_s = 0 \]

\[ \Delta P_s = C W_s + D \] (C and D are obtained from Step 3)

\[ \Delta P_p = A_0 + A_1 W_p + A_2 W_p^2 + A_3 W_p^3 + A_4 W_p^4 \]

The solution occurs when

\[ \Delta P_s = \Delta P_p \]

Then the equation for \( \dot{W}_N \) is

\[ (A_0 - D) + (A_1 - C) \dot{W}_N + A_2 W_N^2 + A_3 W_N^3 + A_4 W_N^4 = 0 \]

Step 5: Solve the equation for \( \dot{W}_N \) using the Newton-Raphson Method of solution for a fourth order polynomial

The remaining steps are identical to that given in Section 2.2.5.1.
3.0 MODIFICATIONS TO SINDA SUBROUTINES

3.1 Preprocessor Modifications

Subroutine IMBED was written to convert actual conductor numbers, node numbers, array numbers or constant numbers which are input in the array data to their relative location in the G, T, A or K arrays respectively. The number to be converted in the array data is entered with an * followed by G, T, A, or K depending on the type of relative location desired. For instance, an array with the following input


would be changed so that the location of A10 in the A array would replace *A10 in the A12+1 location. The relative node number of actual node 5 would replace the *T5 and the relative conductor number of actual conductor number 101 would be placed in A12+3. The converted array might read

12, 102, 3, 21, END

Where A10 is located at location 102 in the A array, actual node 5 has relative number 3, and actual conductor 101 has relative conductor 21.

Subroutine IMBED is called from CODERD. Listings of IMBED, CODERD, and a modified overlay are supplied in Appendix A.

3.2 User Subroutine Modifications

A modification to the SINDA execution subroutine, CNFAST, was required to make it compatible with the radiant interchange subroutine, RADIR. This minor modification was required because the manner in which temperatures were calculated when the problem time increment is larger than the maximum convergence criterion for a given node was not compatible with the manner in which the convergence criteria information was carried over to CNFAST. A listing of the modified version of CNFAST is supplied in Appendix A.

A modification was made to user subroutine TPRINT to print temperatures in increasing order of actual node numbers. This modification increased the dynamic storage requirement to two locations for each node in the network.
4.0 USER SUBROUTINES

The capabilities described in Section 2.0 are available to the SINDA user through user subroutines which were added to the existing SINDA library. This section presents a description and user input requirements for each subroutine. Table 1 summarizes the subroutines and the page that each user description is found.

The subroutine inputs rely heavily upon the capability to convert from actual array, constants, node, and conductor numbers to relative numbers in the array data. To use this capability the user may supply an actual array number, node number or conductor number by preceding the actual number with *A, *T, or *G respectively. This causes the preprocessor to replace the entry with the relative number. Consider the example for array number 2 shown below.


In this example, following the preprocessor phase, *A14 will be replaced by the location in the A array of the Array No. 14 data, *T5 will be replaced by the relative node number for actual node No. 5, and *G7 will be replaced by the relative conductor number for actual conductor No. 7. This feature is used extensively for the input to user subroutines described below and is described in more detail in Section 3.1.
# TABLE 1

**NEW SINDA USER SUBROUTINES**

<table>
<thead>
<tr>
<th>A. Convection Conductor Calculation Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONV1</td>
<td>48</td>
</tr>
<tr>
<td>CONV2</td>
<td>50</td>
</tr>
<tr>
<td>CONV3</td>
<td>52</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B. Flow Conductor Calculation Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOCN1</td>
<td>53</td>
</tr>
<tr>
<td>FLOCN2</td>
<td>53</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C. Time Variant Conductor Calculation Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONDT1</td>
<td>55</td>
</tr>
<tr>
<td>CONDT2</td>
<td>55</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D. Enclosure Radiation Analysis Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RADIR</td>
<td>57</td>
</tr>
<tr>
<td>RADSOL</td>
<td>59</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>E. Heat Exchanger and Heater Analysis Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HXEFF</td>
<td>61</td>
</tr>
<tr>
<td>HXCNT</td>
<td>62</td>
</tr>
<tr>
<td>HXCROS</td>
<td>63</td>
</tr>
<tr>
<td>HXPAR</td>
<td>65</td>
</tr>
<tr>
<td>HEATER</td>
<td>66</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>G. Cabin Analysis Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CABIN</td>
<td>67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>H. Pressure/Flow Balance Subroutine</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFCS</td>
<td>71</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>I. Input/Output Subroutines</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSTRY</td>
<td>83</td>
</tr>
<tr>
<td>NEWTMP</td>
<td>85</td>
</tr>
<tr>
<td>FLPRNT</td>
<td>86</td>
</tr>
<tr>
<td>GENOUT</td>
<td>86</td>
</tr>
<tr>
<td>GENI</td>
<td>86</td>
</tr>
<tr>
<td>GENR</td>
<td>86</td>
</tr>
<tr>
<td>FLUX</td>
<td>87</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>J. Elapsed Computer Time Comparison and Termination Routine</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMCHK</td>
<td>89</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>K. Interpolation</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>REVPOL</td>
<td>90</td>
</tr>
</tbody>
</table>
SUBROUTINE NAME: CONV1

PURPOSE:

CONV1 calculates conductor values using the relations for convection heat transfer for flow in a tube. The relations used to obtain the film coefficient, h, are given by equations 2 thru 4 in section 2.1.1, depending on the flow regime (laminar, transitional or turbulent flow). h is then multiplied by the input area for heat transfer to obtain the conductor value. The conductor value is stored in the input conductor location. Any number of conductors may be calculated with a single call. The flowrate array and the fluid properties data are addressed to tie the convection calculations with the pressure-flow solution. The first argument, AFLOW, is the first argument in the PFCS routine and identifies the flowrate array and fluid type data array needed by CONV1. APR is an array which references the specific heat, density, viscosity and thermal conductivity array. ADAT supplies other information needed.

RESTRICTIONS:

Should be called in the VARIABLES 1 block so that hA values are obtained every iteration.

CALLING SEQUENCE:

CONV1 (AFLOW, APR, ADAT)

where:

AFLOW is first argument of the PFCS call and is of the following format:
AFLOW(IC), AW, AP, AGF, AVP, AIFR, AFT, AFR, APD, END

AW array of flowrates per tube in the system

AP array of pressures per pressure node in the system

AGF array of pressure conductors per tube

AVP array of valve positions for all valves in the system

AIFR array of imposed flowrate per node

AFT array of fluid type data

AFR array of user added flow resistance per tube

APD array for output of pressure drops per tube

APR is the second argument of CONV1 and is also an array in the PFCS data input which contains fluid properties.

It is of the following format: APR(IC), CP, RO, MU, KT, GC, END
CP - is a doublet temperature dependent specific curve when input with the *A format
- is a constant specific heat valve if input as a real constant
RO - is a doublet temperature dependent density curve when supplied using *A
- is a constant density valve when supplied as a real constant
MU - is a doublet temperature dependent viscosity curve when supplied using *A
- is a constant viscosity valve when supplied as a real constant
KT - is a doublet temperature dependent thermal conductivity curve when supplied using *A
- is a constant conductivity valve when supplied as a real constant
GC - is the gravitational constant in the problem under consideration
ADAT - is an argument to CONV1 which contains convection information. It has the following format:
ADAT(IC), NG1, AHT1, ITUBE1, NFL1, ITYPE1, X1, F11, F21
  | | | | | | | |
  | | | | | | | |
  NGn, AHTn, ITUBEn, NFLn, ITYPEn, Xn, F1n, F2n
END

NGi - is actual conductor number of the ith set of data
AHT - is the area for heat transfer
ITUBE is the tube number for obtaining flowrate
NFL - is the actual fluid lump number
ITYPE is the fluid lump type number
X - is the entry length
F1 - is the laminar fully developed coefficient
F2 - is the laminar entry length coefficient
SUBROUTINE NAME: CONV2

PURPOSE:

CONV2 calculates convection conductor values from a user input curve of Stanton number (St) vs Reynolds number (Re). The film heat transfer coefficient is obtained by (1) interpolating a curve of $St\cdot Pr^{2/3}$ versus $Re$ to obtain $St\cdot Pr^{2/3}$ and (2) using the relation $h = k/D \cdot (St\cdot Pr^{2/3}) \cdot Re\cdot Pr^{1/3}$ to obtain $h$. $h$ is multiplied by the heat transfer area to obtain the conductor values which is stored in the proper conductor location. The flow data used by subroutine PFCS is referenced by the arguments AFLOW and APR to obtain flowrate, type data and fluid property data thus tying the convection and flow analysis together.

RESTRICTIONS:

Should be called in the VARIABLES 1 block so that hA values are calculated on each iteration prior to the temperature calculation.

CALLING SEQUENCE:

CONV2(AFLOW, APR, ADAT)

where:

AFLOW - is the first argument of the PFCS call and is of the following format:

AFLOW(IC), AW, APR, AGF, AVR, AIFR, AFT, AFR, APD, END

AW - array of flowrates per tube in the system

APR - array of pressures per pressure node in the system

AGF - array of pressure conductors per tube

AVP - array of valve positions for all valves in the system

AIFR - array of imposed flowrate per node

AFT - array of fluid type data

AFR - array of user added flow resistance per tube

APD - array for output of pressure drops per tube

APR - is an array in the PFCS data input which contains fluid properties.

It is of the following format:

APR(IC), CP, RO, MU, KT, GC, END

CP - is a doublet temperature dependent specific curve when input with the *A

- is a constant specific heat valve if input as a real constant
RO - is a doublet temperature dependent density curve when supplied using *A
- is a constant density value when supplied as a real constant
MU - is a doublet temperature dependent viscosity curve when supplied using *A
- is a constant viscosity value when supplied as a real constant
KT - is a doublet temperature dependent thermal conductivity curve when supplied using *A
- is a constant conductivity value when supplied as a real constant
GC - is the gravitational constant in the problem under consideration
ADAT - is an argument to CONV1 which contains convection information.
It has the following format:
ADAT(IC), NG1, AHT1, ITUBE1, NFL1, ITYPE1, AHST1
    NGn, AHTn, ITUBE n, NFL n, ITYPE n, AHST n
END
NGi - is conductor number of the ith set of data
AHT - is the area for heat transfer
ITUBE - is the tube number for obtaining flowrate
NFL - is the fluid lump number
ITYPE - is the fluid lump type number
AHST - is a doublet curve of ST(PR)\(^{2/3}\) vs Re
**SUBROUTINE NAME:**  CONV3

**PURPOSE:**

CONV3 calculates convection conductor values by interpolating a user supplied curve of heat transfer coefficient, \( h \), versus tube flowrate. The conductor is then obtained by multiplying \( h \) times the area, \( A \). A large number of conductors may be processed with a single call to CONV3. The flow data used by subroutine PFCS is referenced by the argument AFLOW to obtain flowrate and type data.

**RESTRICTIONS:**

CONV3 should be called from the VARIABLES 1 block to obtain updated \( hA \) values on each iteration.

**CALLING SEQUENCE:**

```
CONV3 (AFLOW, ADAT)
```

where:

- **AFLOW** - is the first argument of the PFCS call and is of the following format:
  
  `AFLOW(IC), AW, APR, AGF, AVR, AIFR, AFT, AFR, APD, END`

- **AW** - array of flowrates per tube in the system
- **APR** - array of pressures per pressure node in the system
- **AGF** - array of pressure conductors per tube
- **AVP** - array of valve positions for all valves in the system
- **AIFR** - array of imposed flowrate per node
- **AFT** - array of fluid type data
- **AFR** - array of user added flow resistance per tube
- **APD** - array for output of pressure drops per tube
- **ADAT** - is an argument to CONV1 which contains convection information.
  It has the following format:
  
  `ADAT(IC), NG1, AHT1, ITUBE1, AHW1
  : : : :`
  
  `NGn, AHTn, ITUBEn, AHWn, END`

- **NGi** - is conductor number of the \( i \)th set of data
- **AHT** - is the area for heat transfer
- **ITUBE** - is the tube number for obtaining flowrate
- **AHW** - is a doublet array of heat transfer coefficient vs flowrate
SUBROUTINE NAME: FLOCN1 or FLOCN2

PURPOSE:
Subroutine FLOCN1 and FLOCN2 calculate thermal conductor values required for thermal characterization of fluid flowing down a tube. The conductor values are obtained by multiplying the tube flowrates times the specific heat for each of the conductors identified in the ADAT array. Both subroutines reference the flowrate array, AW, and FLOCN1 references the ACP array which gives the specific heat vs temperature relationship. FLOCN2 assumes a constant value for specific heat.

The conductor values referenced in the ADAT array must also be supplied in the CONDUCTOR DATA block as one-way conductors with the proper connections identified. Any dummy value may be supplied for the initial flow conductor values since these values will be replaced following the first call to FLOCN1 or FLOCN2. These subroutines are called from VARIABLES 1.

RESTRICTIONS:
Must be called from VARIABLES 1.

CALLING SEQUENCES:
FLOCN1 (AW, ACP, ADAT1) or FLOCN2 (AW, CP, ADAT2)

where

AW - is the array of flowrates per tube also referenced in subroutine PFCS
ACP - is a doublet array of specific heat versus temperature
Cp - is a constant value of specific heat
ADAT1 - is the array which identifies the conductor, the corresponding upstream lump and the tube number for each conductor. It is of the format:
ADAT1, NG1, UPL1, ITUBE1
    NG2, UPL2, ITUBE2
    . . .
    NGn, UPLn, ITUBE_n, END
ADAT2 - is the array which identifies the conductor, number and the tube number for each conductor. It is of the format:
ADAT2, NG1, ITUBE1, NG2, ITUBE2, --, NGn, ITUBEn, END

NGi - is the ith conductor number (the *G notation is used)

UPLi - is the upstream lump number for conductor number NGi
(The *T notation is used)

ITUBEi - is the tube number which contains the flow for the ith conductor \( \dot{\omega} \). Cp product. For flow splitting or mixing junctions, ITUBEi should be the number of the connect tube containing the smallest amount of flow. For example, for a splitting junction the flow conductor which crosses the junction should contain the downstream tube. For a mixing junction ITUBE should be the upstream tube.
SUBROUTINE NAME: CONDT1 or CONDT2

PURPOSE:
Subroutines CONDT1 and CONDT2 calculate the value of conductance values as a product of a time variant argument W(t) and a temperature variant or constant argument Cp. The value of Cp is temperature dependent with CONDT1 and is constant for CONDT2. The subroutines were written primarily for the purpose of evaluating flow conductors for the case of flowrate a given function of time. However, they may be used for other time variant conductor applications.

RESTRICTIONS:
Should be called from VARIABLES 1.

CALLING SEQUENCE:
CONDT1 (ADAT1) or CONDT2 (ADAT2)
where ADAT1 is of the form
ADAT1, NG1, NLT1, ATIME1, ATEMP1
   NG2, NLT2, ATIME2, ATEMP2
     ... ... ...
   NGn, NLTn, ATIMEn, ATMEPn, END
and ADAT2 is of the form:
ADAT2, NG1, ATIME1, CP1
   NG2, ATIME2, CP2
     ... ... ...
   NGn, ATIMEn, CPn, NED

The following definitions apply to the above.

NGi - the conductor number of the ith conductor addressed in ADAT1 or ADAT2. The *G notation should be used.

NLTi - the lump whose temperature will be used to interpolate the ATEMPi array to obtain the Cp constant. The *T notation should be used.

ATIMEi - the time variant array for determining the value of W(t) for the ith conductor. The *A notation should be used.
ATEMPi - the temperature variant array which is interpolated with the temperature of lump NLTi to obtain Cp in CONDT1. The conductor is calculated as the value of W(t)*Cp. The *A notation should be used.

Cp - the constant value which will be multiplied by W(t) from the ATIMEi array to obtain the conductor value for CONDT2.
SUBROUTINE NAME: RADIR

PURPOSE:
RADIR calculates the script-F values for infrared radiation heat transfer within an enclosure and uses these values to obtain the heat transfer during the problem. Several temperature nodes may be combined on a single surface for radiation heat transfer purposes. Also, the user may analyze problems with specular, diffuse or combinations of specular and diffuse radiation. See Section 2.1.6.1 for definitions and detailed description of methods.

RADIR calculates the script-F values on the initial call. This is performed by the procedure outlined in Section 2.1.6.1, Equations 23, 25 and 31. These values replace the EFT values in the SC array for future use. The heat flux values are then calculated on all iterations by:

1. Calculating the temperature of each surface using equation 34
2. Calculating the absorbed heat for each node by the relation of equation 33

The value given by equation 35 is added to the conductor sum for each node so that the proper convergence time increment may be obtained. As many enclosures as desired may be analyzed by each enclosure but each enclosure requires a different call to RADIR. RADIR must be called in VARIABLES 1.

RESTRICTIONS:
Must be called from VARIABLES 1 Surface nodes must be boundary nodes

CALLING SEQUENCE:
RADIR (A(IC)).

Where A is of the following format:
A(IC),SN,SE,SR,SC,NA,SP,END

SN,SE,SR,SC,NA, and SP are actual array numbers input using the *A procedure and are of the following formats

SN(IC),n,SN1,SN2,SA2,NN2,..............SNn,SAa,NNn,END
SE(IC),SE1,SE2-----SEn,END
SR(IC),SR1,SR2-----SRn,END
SC(IC),SNF1,SNF2,SNT2,EFT2,---SNFm,SNTm,EFTm,END
NA(IC),NNO(1,1),AN(1,1),NNO(1,2),AN(1,2)--NNO(1,NN1),AN(1,NN1)
NNO(2,1),AN(2,1),NNO(2,2),AN(2,2)--NNO(2,NN2),AN(2,NN2)
NNO(n,1),AN(n,1),NNO(n,2),AN(n,2)--NNO(n,NNn),AN(n,NNn),END
SP(IC),SPACE,NSPACE,END
The following definitions apply in the above calling sequence:

A        Array identification for the array which identifies the other arrays containing the data
SN       Array number for the array containing surface numbers and areas
SE       Array number for the array containing the surface emissivities
SR       Array number for the array containing the surface reflectivities
SC       Array number for the array containing the surface connections data
NA       Array number for the array containing the temperature node numbers and areas
SP       Array number for the array containing the space which is used for obtaining script FA values and for subsequent temperature calculations

The number of surfaces

SN1,SN2,...SNn  Node number for surfaces - must be boundary nodes
SA1,SA2,...SAN  Total area for each surface
NN1,NN2,...NNn  Number of temperature nodes on each surface
SE1,SE2,...SEN  Emissivity values for each surface
SR1,SR2,...SRn  Diffuse reflectivity values for each surface
SNF1,SNT1,EFT1  Connections data: Surface number from, surface number to, E value from SNF1 to SNT1, etc.

NNO(X,Y)  Temperature node numbers on surfaces; Node number Y on surface X
AN(X,Y)   Area of node Y on surface X
NSPACE   Number of spaces needed to store script-FA values - NSPACE must be an integer value of n*n(n+1)/2

m        The number of surface connections
SUBROUTINE NAME: RADSOL

PURPOSE:

RADSOL calculates a pseudo script-F for radiation from an external source entering an enclosure and uses these values to calculate the net heat transfer to each node due to the entering source. A number of temperature nodes may be combined on a single surface for radiation purposes. Also, problems with specular, diffuse, or combinations of specular and diffuse radiation may be analyzed. Section 2.1.6.2 should be consulted for definitions and descriptions of methods.

RADSOL calculates the pseudo script-F values on the initial call. This is performed by equations 38, 40, and 44 of section 2.1.6.2. The values are stored in the EFT values of the SC array supplied by the user. The heat flux values are then calculated on each iteration by equations 45 and 46.

The user may analyze as many enclosures as desired by supplying a call statement for each enclosure. Also, a user may analyze several wave length bands by supplying a call to RADSOL for each wave length band.

RESTRICTIONS:

Must be called from VARIABLES I; Surface nodes must be boundary nodes.

CALLING SEQUENCE:

RADSOL (A(IC))

Where the A array is of the following format:

A(IC),SN,SE,SR,HT,SC,NA,SP,END

SN,SE,SR,HT,SC,NA, and SP are actual array numbers input using the *A procedure and are of the following formats:

SN(IC), n,SN1,SA1,NN1,SN2,SA2,NN2,----Snn,SA n,NNn,END
SE(IC),SE1,SE2,-----SE n,END
SR(IC),SR1,SR2,-----SR n,END
HT(IC),HT1,HT2,-----HT n,END
SC(IC),SCF1,SCNT1,SCF2,SCNT2,SCF2,SCFT1,SCFT2,SCFTm,SCFTm,END
NA(IC),NNO(1,1),AN(1,1),NNO(1,2),AN(1,2)---NNO(1,NN1),AN(1,NN1),
NNO(2,1),AN(2,1),NNO(2,2),AN(2,2)---NNO(2,NN2),AN(2,NN2),
\vdots \vdots \vdots \vdots \vdots \vdots \vdots
NNO(n,1),AN(n,1),NNO(n,2),AN(n,2)---NNO(n,NNn),AN(n,NNn),END
SP(IC),SPACE,NSPACE,END
The following definitions apply in the above calling sequence

A Array identification for the array which identifies the other arrays containing the data

SN Array number for the array containing surface numbers and areas

SE Array number for the array containing the surface emissivities

SR Array number for the array containing the surface reflectivities

HT Array number for the array containing the incident heat curves or constant heat flux values

SC Array number for the array containing the surface connections data

NA Array number for the array containing the temperature node numbers and areas

SP Array number for the array containing the space which is used for obtaining script values and for subsequent temperature calculations

SN1,SN2,...SNn Node number for surfaces; must be boundary nodes

SA1,SA2,...SAN Total area for each surface

NN1,NN2,...NNn Number of temperature nodes on each surface

SE1,SE2,...SEN Emissivity values for each surface

SR1,SR2,...SRn Diffuse reflectivity values for each surface

SHT1,SHT2,...SHTn Incident heat flow on surfaces; may identify curves containing incident values vs time

SNFl,SNT1,EFTI Connections data: Surface number from surface number to, E value from SNFl to SNT1, etc.

NNO(X,Y) Temperature node numbers on surfaces: Node number Y on surface X

AN(X,Y) Area of node Y on surface X

NSPACE Number of spaces needed to store script-FA values - NSPACE must be an integer values of \( n/2(n+1) \)
SUBROUTINE NAME:          HXEFF

PURPOSE:

This subroutine obtains the heat exchanger effectiveness either from a user constant or from a biariant curve of effectiveness versus the flow rates on the two sides. The effectiveness thus obtained is used with the supplied flow rates, inlet temperatures and fluid properties to calculate the outlet temperatures using the methods described in Section 2.1.3.4. The user may specify a constant effectiveness by supplying a real number or may reference and array number to specify the effectiveness as a biariant function of the two flow rates. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXEFF should be called in the VARIABLES 1 block. The value for EFF, the first argument must never be zero. Tout1 and Tout2 must be boundary nodes.

CALLING SEQUENCE:

HXEFF(EFF,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2)

Where EFF - is (1) the effectiveness if real, (2) a curve number of a biariant curve of effectiveness versus W1 and W2 if an array

W1,W2 - are the flow rates for side 1 and 2 respectively. May reference the flow rate array, AW+I where I is the tube number

CP1,CP2 - are the specific heat value for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties

TIN1,TIN2 - are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2

TOUT1,TOUT2 - are the outlet lump temperature locations sides 1 and 2 where the calculated values will be stored Must be boundary nodes
SUBROUTINE NAME: HXCNT

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relation described in Section 2.1.3.1 for a counter flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array number. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXCNT should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. Tout1 and Tout2 must be boundary nodes.

CALLING SEQUENCE:

HXCNT(UA,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2)

where UA
is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array

W1,W2
are the flow rates for side 1 and side 2 respectively. May reference the flow rate array, AW+1 where is the tube number

CP1,CP2
are the specific heat values for side 1 and 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties

TOUT1-TOUT2
are the outlet lump temperature locations (sides 1 and 2) where the calculated values will be stored Must be boundary nodes

62
SUBROUTINE NAME: HXCROS

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relations described in Section 2.1.3.3 for a cross flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array number. Any one of the following four types of cross flow exchangers may be analyzed (see Section 2.1.3.3 for the relations):

1) Both streams unmixed
2) Both streams mixed
3) Stream with smallest MCp product unmixed
4) Stream with largest MCp product unmixed

The type is specified by the last argument in the call statement. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for both sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXCROS should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. Tout1 and Tout2 must be boundary nodes.

CALLING SEQUENCE:

HXCROS(UA,W1,W2,CP1,CP2,T1N1,T1N2,TOUT1,TOUT2,K)

Where

UA is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array.

W1,W2 are the flow rates for side 1 and side 2 respectively. May reference the flow rate array, AW+I where I is the tube number.

CP1,CP2 are the specific heat values for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties.

T1N1,T1N2 are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2.

TOUT1,TOUT2 are the outlet lump temperature locations(sides 1 & 2) where the calculated values will be stored. Must be boundary nodes.
$K$ is the code specifying type of cross flow exchanger:

- Both streams unmixed: $K=1$
- Both streams mixed: $K=2$
- Stream with small WCp Unmixed: $K=3$
- Stream with large WCp Unmixed: $K=4$
SUBROUTINE NAME: HXPAR

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relations described in Section 2.1.3.2 for a parallel flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXPAR should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. T_{out1} and T_{out2} must be boundary temperatures.

CALLING SEQUENCE:

HXPAR(UA,W1,W2,CP1,CP2,TIN1,TIN2,OUT1,TOUT2)

Where

UA is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array.

W1,W2 are the flow rates for side 1 and 2 respectively. May reference the flow rate array,AW+I where I is the tube number.

CP1,CP2 are the specific heat values for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperatures dependent curves.

TIN1,TIN2 are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2.

TOUT1,TOUT2 are the outlet lump temperature locations (sides 1 and 2) where the calculated values will be stored (should be boundary temperatures).
SUBROUTINE NAME: HEATER

PURPOSE:

This subroutine simulates an electrical heater with a control system which turns the heater on when the sensor lump temperature falls below the "heater on" temperature TON, and turns the heater off when the sensor lump rises above the heater off temperature, TOFF. When the heater is on, the input Q value is added to the Q location specified by the user. When the heater is off, no heat is added.

RESTRICTIONS:

HEATER must be called in the VARIABLES 1 block.

CALLING SEQUENCE:

HEATER(Q,QHT,KODE,TSEN,TON,TOFF)

Where

TSEN is the sensed temperature
TON is the heater on temperature
TOFF is the heater off temperature
QHT is the heater heat rate
Q is the location for storing the heat

KODE is an integer variable set by HEATER
= 1 if the heater was "on" at last call
= 0 if the heater was "off" at the last call
(User sets KODE for first call)
SUBROUTINE NAME: CABBIN

PURPOSE:

This subroutine performs a thermal and mass balance on a cabin air system. The cabin air is assumed to be a two component gas mixture with one condensible component and one noncondensible component. The cabin air is assumed to be well mixed so that the temperature and specific humidity are constant throughout. The cabin may contain any number of entering streams each with different temperature and humidity conditions. The cabin air may transfer heat to any number of nodes in its surroundings with the heat transfer coefficient obtained by one of the three options:

1. User input coefficient
2. Relations for flow over a flat plot
3. Relations for flow over a tube bundle

The relations describing the second and third options are given in Section 2.1.5. The mass transfer coefficient for determining the rate of condensation or evaporation is determined by the Lewis relation which relates the mass transfer coefficient directly to the convection heat transfer coefficient. By the Lewis Relation, if the diffusion coefficient is approximately equal to the thermal diffusivity, the Sherwood number is approximately equal to the Nusselt number, thus giving a direct relation. (See Section 2.1.5 for details). Mass and heat transfer rates are determined at each node that interfaces the cabin gas as well at entering and exiting streams and a new cabin gas temperature and humidity is determined each iteration based upon the heat and mass balance. An account is kept of the condensate on the walls when condensation occurs but the condensate is assumed to remain stationary and not flow to other wall nodes.

Limits are applied when necessary to prevent more condensation than the vapor existing under severe transient condition and to prevent evaporation of more liquid than exists at each wall lump.

As many cabins as desired may be analyzed in a given problem, but each must contain separate input information.

RESTRICTIONS:

CABIN must be called in VARIABLES 1.

CALLING SEQUENCE:

CABIN(A(IC) TC, TC, K1, K2)

The following definitions apply to the above calling sequence:

A is an array containing arrays numbers which contain cabin input information.
The cabin gas temperature which must be a boundary node

Storage locations needed by CABIN

The A array has the following format where the *A procedure is used:

A(IC),IF,PR,CN,H,FP,TB,SP,END

Where IF Identifies an array containing the entering flow rate information. The format of the array is:

IF(IC),NS,FR_1,PSI_1,FR_2,PSI_2,TE_2-------FR ns,PSI ns,TE ns

PR Identifies an array identifying array numbers for property values. The format of the array is:

PR(IC),NFLC,NMUO,NMUV,NCPV,NKO,NKV,NLAT

CN Identifies an array containing pertinent constants. The format of the array is:

CN(IC),RA,RV,VC,PC,XC,WV,PSIC,PO,TO,CONV

H Identifies an array containing node numbers and convection heat transfer coefficient values for nodes surrounding the cabin gas. The format of the array is:

H(IC),LN_1,HA_1,LN_2,HA_2,---LN_n1,HA_n1

FP Identifies an array containing node numbers and information to permit calculation of convection coefficients for flat plates. The format is:

FP(IC),LN_1,XX_1,XI_1,VIW_0_1,LN_2,XX_2,XI_2,VIW_0_2

TB Identifies an array containing node numbers and information to permit calculation of convection coefficients for tube bundles. The format is:

TB(IC),LN_1,DI_1,VIW_0_1,LN_2,DI_2,VIW_0_2,------LN_n3

DI_n3,VIW_0_n3

SP Identifies an array which contains working space equal to or greater than three times the sum of the number of nodes with input heat transfer coefficients plus the number using flat plot relations plus the number using tube bundles.

The following symbol definitions apply in the above:
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS</td>
<td>Number of incoming streams</td>
</tr>
<tr>
<td>FR&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Entering flow rate for stream i</td>
</tr>
<tr>
<td>PSI&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Specific humidity for entering stream i</td>
</tr>
<tr>
<td>TE&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Temperature of entering stream i</td>
</tr>
<tr>
<td>NFLC</td>
<td>Curve number for circulation flow rate vs time</td>
</tr>
<tr>
<td>NMUO</td>
<td>Curve number for noncondensible viscosity vs temperature</td>
</tr>
<tr>
<td>NMUV</td>
<td>Curve number for condensible viscosity vs temperature</td>
</tr>
<tr>
<td>NCPO</td>
<td>Curve number for noncondensible specific heat vs temperature</td>
</tr>
<tr>
<td>NCPV</td>
<td>Curve number for condensible specific heat vs temperature</td>
</tr>
<tr>
<td>NKO</td>
<td>Curve number for noncondensible thermal conduction vs temperature</td>
</tr>
<tr>
<td>NKV</td>
<td>Curve number for condensible thermal conduction vs temperature</td>
</tr>
<tr>
<td>NLAT</td>
<td>Curve number for latent heat of condensible vs temperature</td>
</tr>
<tr>
<td>RA</td>
<td>Gas constant for non-condensible component</td>
</tr>
<tr>
<td>RV</td>
<td>Gas constant for condensible component</td>
</tr>
<tr>
<td>VC</td>
<td>Cabin volume</td>
</tr>
<tr>
<td>PC</td>
<td>Cabin Pressure</td>
</tr>
<tr>
<td>XC</td>
<td>Molecular weight ratio, Mv/Mo</td>
</tr>
<tr>
<td>WV</td>
<td>Initial vapor weight in cabin</td>
</tr>
<tr>
<td>PSIC</td>
<td>Initial specific humidity for cabin</td>
</tr>
<tr>
<td>LN&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Cabin wall lump</td>
</tr>
<tr>
<td>HA</td>
<td>Heat transfer coefficient times area</td>
</tr>
<tr>
<td>n1</td>
<td>Number of wall lumps which have input HA values</td>
</tr>
<tr>
<td>n2</td>
<td>Number of wall lumps which have HA calculated by flat plate relations</td>
</tr>
<tr>
<td>n3</td>
<td>Number of wall lumps which have HA calculated by tube bundle relations</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>xx&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Distance from leading edge for flat plate heating for ith flat plate node</td>
</tr>
<tr>
<td>XI&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Length of flat plate in flow direction for ith flat plate node</td>
</tr>
<tr>
<td>AI&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Heat transfer area for flat plate or tube node</td>
</tr>
<tr>
<td>DI&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Tube outside diameter for tubes in the bundle for ith tube node</td>
</tr>
<tr>
<td>VIWO</td>
<td>Ratio of velocity at the lump to the circulation flow rate</td>
</tr>
<tr>
<td>To</td>
<td>The reference temperature to be used for estimating the saturation pressure of the condensible component. Should be near the range of saturation temperature expected</td>
</tr>
<tr>
<td>Po</td>
<td>The saturation pressure at To for the condensible component</td>
</tr>
<tr>
<td>CONV</td>
<td>Conversion factor to make the quantity XLAM/Rv/To dimensionless where XLAM is the latent heat of vaporization and Rv is the gas constant for the vapor. If XLAM is BTU/lb, Rv is FT-LB/°R and To is °R, CONV=778.</td>
</tr>
</tbody>
</table>
SUBROUTINE NAME: PFCS

PURPOSE:
Subroutine PFCS determines the flow distribution in a set of general parallel/series fluid flow tubes so that the pressure drop values between any parallel flow paths are equal and flow is conserved. The following effects are included in the pressure drop calculations:

1. pipe flow friction
2. orifices and fittings
3. valves

The effect of temperature dependent properties are included in the calculations. The properties are evaluated at the temperature of each fluid lump in each tube in evaluating the flow resistance when setting up the equations to be solved.

A balance is made between the flow/pressure drop characteristics of the system and the flow/pressure rise of a pump for each system concurrent with the system pressure flow solution to obtain the incoming system flowrate. A detailed discussion of the equations and techniques used are described in Section 2.2.

General flow charts of PFCS and supporting subroutines are shown in Fig. 7, 8, & 9.

RESTRICTIONS:
Must be called from VARIABLES 2. The system of units used for the thermal and flow problems should be consistent.

CALLING SEQUENCE: PFCS (AFLOW, ADAT, NAME)

where AFLOW - is an array which references other arrays for flowrates, pressures, flow conductors, valve positions, imposed flow rates, fluid type data, user added flow resistances and pressure drops. It is of the following format where the *A conversion feature described in Section 3.1 is used to reference arrays.

AFLOW (IC), AW, APN, AGF, AVP, AIFR, AFT, AFR, APD, END

ADAT - is an array which identifies other arrays containing fluid property values, parameters needed for the pressure/flow solution numerical technique, the flow system network, valve data, pump data and a check outprint code. The integer count should be addressed. The format of ADAT is as follows where
the *A format is used to address array values:
ADAT(IC), APR, ASOL, ANET, AVLS, AP, KOP, END

NAME - is an array containing the name of the network (it may also be supplied as a Hollerith using the H format). Nine 6 character words should be used.

AFLOW ARGUMENTS

AW - is the array number of an array containing flowrates per tube for each system. The integer count must be addressed and it must contain the number of spaces exactly equal to the number of tubes in the system.

APN - is an array number for an array containing the pressures for each pressure node in the system. On input the user need only set up the space. The integer count must be addressed and it must contain the number of spaces exactly equal to the number of pressure nodes in the system.

AGF - is an array number for an array containing the flow conductors for each tube in the system. The user needs only to setup the space on input which must be exactly equal to the number of tubes in the flow system.

AVP - is an array number for an array containing valve positions in order of valve numbers. The integer count must be addressed and the number of input valves must be exactly equal to the number of valves. The user supplies the initial valve positions in this array.

AIFR - is the array location of an array of imposed flow sources for each pressure node. The integer count must be addressed and the array must contain the number of spaces exactly equal to the number of pressure nodes in the flow system.

AFT - is the array location of an array which contains fluid lump type data. The AFT array is of the following format:

AFT, WP1, CSA1, FLL1, MFF1, NHL1, FFC1
    , , , , , ,
    , , , , , ,
    WPnt, CSAnt, FLLnt, MFFnt, NHLnt, FFCnt
END

nt - is the number of types
WP_i - is the wetted perimeter of fluid type i
CSAi - is the cross sectional area for fluid type i
FLLi - is the fluid lump length for fluid type i
MFFi - is the curve of friction factor vs Reynolds number for Reynolds number greater than 2000 when greater than 0.
- is a key to use internal calculations methods for friction factor when MFF = 0.
NHLi - is the number of head losses for type i when real
- is the number of an array of head losses vs Reynolds number when an integer
FFCi - is a user input constant to be multiplied times the friction factor to modify it for type i.
AFR - is an array number of an array containing user added flow resistances for the tubes. This can be used to include the effects of changes in flow altitudes or the effects of valve types not available in the valve package or other known flow resistances. The integer count must be addressed and the number of values in the array must be exactly equal to the number of tubes in the flow system.
APD - is an array number of an array which will contain pressure drop values for all tubes in the flow system following a call to PFCS. The array is strictly for output purposes. The integer count must be addressed and the number of array valves must be exactly equal to the number of tubes in the system.

ADAT ARGUMENTS

APR - is an array identifying the fluid properties data and GC, the gravitational constant. It is of the following format: APR, CP, RO, MU, KT, GC, END
CP, RO, MU, KT, are the values of fluid specific heat, density, viscosity, and thermal conductivity respectively or the appropriate array reference (using the *A format). The value is constant for any of the properties if a real number is supplied. The integer count must be referenced when variable properties are used.

73
GC - is the gravitational constant. Table 2 gives the value for various system of units.

ASOL - is an array number of an array containing various numerical solution parameters needed by PFCS. ASOL is of the following format:

ASOL (IC), TOL, MXPASS, EPS, FRDF, END

TOL - is the solution tolerance on rate of change of flowrates from one pass to the next. The fraction of change must be within TOL for all tubes in any system or subsystem before a solution is reached. TOL must be greater than 0. A typical value is 0.001

MXPASS - is the maximum number of passes permitted in the balancing loop of PFCS to obtain a pressure-flow solution on any given iteration. This value should always be greater than 20 with a typical value of 100.

EPS - Not used but a space must be supplied.

FRDF - is the flowrate damping factor used to accelerate the rate of convergence for the iterative solution to the set of nonlinear equations. This value should generally be between 0.5 and 1.0. Values of 0.5 to 0.7 have been found best for most turbulent flow problems.

ANET - is the array number of an array which identifies the tube connections, pressure nodes connected and fluid lumps contained in the tube. The format of ANET is as follows were the *A format is used for AD, APNPS, and AVL:

ANET,NNAME, APNPS, AVL
TUBE1, NFRM1, NTO1, KD1, AD1
TUBE2, NFRM2, NTO2, KD2, AD2
... ...
TUBE_n, NFRM_n, NTO_n, KD_n, AD_n, END
NNAME - is a six character name to be used for identifying the network in output statements, etc.

APNPS - is an array number (referenced using *A) of an array identifying nodes with specified pressures and is of the form
APNPS, NSPR1, NSPR2, - - - - - NSPRn, END

NSPRI - is the ith pressure node with specified value

TUBEi - is the tube number of the ith connection

NFRMi - is the "from" pressure node for the ith connection

NTOi - is the "to" pressure node for the ith connection

KDi - is an integer code to identify the type of conductor for the ith connection. (See ADi below)

ADi - is the data to be used for calculating the conductor value for the ith connection.

If KDi < 0, the conductor value is the equivalent conductance of a subnetwork described by array ADi. ADi is then of identical format to ANET.

If KDi > 0, the conductor is obtained by the normal pressure drop equations and array ADi fluid lumps, fluid lump types and tube lumps that are contained in the tube. The form of ADi is
ADi, NFLMP1, ITYPE1, NTLMP1
    NFLMP2, ITYPE2, NTLMP2
      :    :    :
    NFLMPn, ITYPEn, NTLMPn, END

Where NFLMPi is ith temperature lump contained in the tube, ITYPEi is the NFLMPi fluid lump type, and NTLMP is the tube lump containing NFLMPi.

If KDi = 0, the conductor calculation is not made, allowing the user to supply the pressure conductance value. ADi is not used.
AVLS - is the array location (identified in array ADAT using the *A format) of an array which identifies the array location of the valve data for all the valves in the system.

AVL - is the array location (identified in array ANET using the *A format) of an array which identifies the array locations of the valve data for the valves in the network or subnetwork described by ANET.

AVLS and AVL are of the following format:

```
AVLS or AVL, AVLV1, AVLV2, ----- AVLVNV, END
```

Where AVLVi is the array number (using *A) of an array which contains the valve data for the ith valve in AVLS or AVL. The format for the valve data arrays, AVLVi, is one of three forms depending on the valve type (rate limited, polynomial, or switching).

The format for a rate limited valve is:

```
AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, TSEN1, TSEN2, DB, RF, RL, END
```

The format for a polynomial valve is:

```
AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, TSEN1, TSEN2, AO, A1, A2, A3, A4, A5, VTC, END
```

The format for a switching valve is:

```
AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, NSEN, T1, T2, END
```

The following definitions apply for the above arrays:

- **NV** - Valve number
- **NTS1** - Tube number connected to side 1 of the valve
- **NTS2** - Tube number connected to side 2 of the valve
- **MODE** - Operating mode: 1 - operating; 0 - not operating
- **XMIN1** - Side 1 minimum position; side 2 maximum position is \(1.0 - XMIN1\)
- **XMAX1** - Side 1 maximum position; side 2 minimum position is \(1.0 - XMAX1\)
- **E** - The valve geometric factor relating pressure drop through the valve by

\[ \Delta P = E \left( \frac{\text{flowrate}}{\text{valve position}} \right)^2 \]
TSEN1 - Sensor lump for side 1 or set point for side 2; If TSEN1 is an integer, it identifies the side 1 sensor lump to be controlled to (a) the set point for side 1 or (b) the sensor lump for side 2 (TSEN2). If the variable is input as a real number it represents a set point to which the side 2 sensor lump will be controlled.

TSEN2 - Sensor lump for side 2 or set point for side 1; If TSEN2 is an integer, it identifies the side 2 sensor lump to be controlled to (a) the set point for side 2 or (b) the sensor lump for side 1 (TSEN1). If the variable is input as a real number it represents a set point to which the side 1 sensor lump will be controlled.

AO, A1, A2, A3, A4, A5 - Polynomial curve fit coefficients for a curve fit of the steady state valve position vs sensed temperature error for side 1:

\[ X_{\text{SS}} = A0 + A1 \cdot \Delta T + A2 \cdot \Delta T^2 + A3 \cdot \Delta T^3 + A4 \cdot \Delta T^4 + A5 \cdot \Delta T^5 \]

DB - Dead band for the rate limited valve, degrees of temperature (See Figure 5).

RF - Rate factor, the rate of change of valve velocity to sensed temperature error (dx/d(\Delta T)) as shown on Figure 5.

RL - Rate limit, the maximum valve velocity, \( \dot{x}_{\text{max}} \) (See Figure 5).

VTC - Valve time constant as described in Section 2.2.3.2. If a valve is desired with no time lag, a time constant which is very small compared to the problem time increment should be input. (VTC must be greater than zero).

NSEN - Sensor lump for switching valve

T1 - Side 1 off temperature or side 2 on temperature for switching valve

T2 - Side 2 off temperature or side 1 on temperature for switching valve

AP - is the array number of an array containing the pump data for the system specified in the ADAT array using the *A nomenclature. The format of the AP array is different for different types of pumps. If flowrate is a function of time the format is (where AW is supplied using *A):

77
If the flowrate is obtained using a tabulated pump curve the format is:
(Where ADP is supplied with *A)
AP, NPI, NPO, ADP, END

If the flowrate is obtained using a polynomial pump curve, the format is:
AP, NPI, NPO, AO, A1, A2, A3, A4, END

The following definitions apply in the above arrays:
NPI - System inlet pressure node
AW - Tabulated curve of flowrate vs time
NPO - System outlet pressure node
ADP - Tabulated pump curve giving pressure rise as a function of flowrate
AO, A1, A2, A3, A4 - Polynomial curve fit constants for flowrate as a function of pressure rise. i.e.,
\[ \dot{w} = A0 + A1 \cdot \Delta P + A2 \cdot \Delta P^2 + A3 \cdot \Delta P^3 + A4 \cdot \Delta P^4 \]

KOP - is an integer code for checkout print from subroutine PFCS.
If KOP = 1 a checkout print will be obtained. If KOP = 0 a print will not be obtained.

DYNAMIC STORAGE REQUIREMENTS:

Dynamic storage required for PFCS is 1/2(NPRN^2 + 6*NPRN+12), where NPRN is the maximum of the number of pressure nodes in any network.
<table>
<thead>
<tr>
<th>UNITS</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASS</td>
<td>FORCE</td>
</tr>
<tr>
<td>LB</td>
<td>LBf</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FIGURE 7 FLOW CHARTS OF PFCS AND NTWRK
FIGURE 8  FLOW CHARTS OF NTWRK1 AND NTWRKN
FLOBAL

CALCULATE VALVE RESISTANCES

SET UP SPACE FOR MATRIX, RIGHT HAND SIDE, ETC.

ASSEMBLE COEFFICIENT MATRIX

STORE IMPOSED FLOW RATES IN RHS

STORE INLET FLOW RATE IN RHS

REDUCE MATRIX AND ADJUST RHS

CALL SYMSOL FOR PressURES

A

CALCULATE NEW FLOW RATES

RETURN

FIGURE 9 FLOW CHART OF FLOBAL
SUBROUTINE NAME: HSTRY

PURPOSE:

Subroutine HSTRY stores the problem time, the pressures of all pressure nodes, the valve positions for all valves, the flowrates for all tubes, and the temperatures of all temperature nodes at an input interval on a magnetic tape (the history tape) mounted on Unit T. The number of records written on the history tape is the number of history intervals plus two. The first record contains a title, an integer count of the number of items to be written for each of the four categories (pressures, valve positions, flowrates and temperatures), and the actual node numbers in order of the relative numbers. The second thru the next-to-last records contain the history records with one for each time point and the last record is the same as the next-to-last except the time is negative. The arguments to HSTRY are the pressure array, PR, the valve position array, VP, the flow rate array, W, and the history tape writing interval, TINC.

The format for the history tape is as follows:

Record No. 1
Title (Written Internally) in 12A6 format, 0, 0, 0, 0, 0, 0, No. of pressure nodes, number of valve positions, 0, 0, 0, number of tubes, 0, 0, number of nodes, actual node numbers in increasing order of relative node numbers.

Record No. 2
Initial problem time, pressures, valve positions, flowrates, node temperatures

Record No. 3
Second history time, pressures, valve positions, flowrates, node temperatures

Record No. N+1 (Where N = number of history time slices to be written)
Last history time, pressures, valve positions, flowrates, node temperatures
Record No. N+2
Same as last record except time is negative

RESTRICTIONS:
Should be called in VARIABLES 2. An output history tape should be mounted on Unit T. Subroutine TMCHK must be in VARIABLES 2 prior to the call to Subroutine HSTRY if TIMCHK is called in the problem.

If the backup feature is used in VARIABLES 2, the call to subroutine HSTRY should not be made until the last pass to avoid nonincreasing time records or invalid data. For example:

```
BCD 3VARIABLES 2

F IF (T(16) .LT. TMAX) BACKUP = 1.

F IF (BACKUP .GT. 0.) GO TO 10
HSTRY (A1, A2, A3, .01)
F 10 CONTINUE
END
```

CALLING SEQUENCE:
```
HSTRY(PR(IC), VP(IC), W(IC), TINC)
```

- PR - is the pressure (or pressure drop) array
- VP - is the valve position array
- W - is the flowrate array
- TINC - is the time interval for plotting
SUBROUTINE NAME: NEWTMP

PURPOSE:

Subroutine NEWTMP will read the node temperatures, flowrates, pressures and valve positions at time TMPTIM from the history tape assigned to Unit U generated by subroutine HSTRY for a previous run on Unit T to initiate a problem at these conditions. The pressure array, PR, valve position array, VP, flow rate array, W, and time to read the tape, TMPTIM, are arguments. The subroutine should be called in the execution block prior to the call to the temperature solution subroutine.

REstrictions:

Must be called in the EXECUTION block prior to the call to the appropriate temperature solution subroutine. The history tape must be assigned on Unit U.

Calling sequence:

NEWTMP(PR(IC), VP(IC), W(IC), TMPTIM)

PR - is the pressure array
VP - is the valve position array
W - is the flowrate array
TMPTIM - is the time to read the values of PR, VP, W and temperatures from the U tape
SUBROUTINE NAME: FLPRNT

PURPOSE:
Subroutine FLPRNT will write the values of the DATA array of real numbers at 10 to a line. The array is labeled by the variable input HEAD which contains 9 six character alpha numeric words. The array location of every tenth value in the array is identified to the right of the appropriate line. FLPRNT was written primarily for the output of flowrates, pressures, pressure drops, and valve positions obtained from PFCS but may be used for the output of any real array.

RESTRICTIONS:
Should be called from OUTPUT. The array must be real.

CALLING SEQUENCE:
FLPRNT(DATA(IC), HEAD(DV))

SUBROUTINE NAMES: GENOUT, GENI OR GENR

PURPOSE:
These subroutines print out arrays of numbers 10 to a line. GENOUT prints either real numbers, integer or both. GENI and GENR print integers and real number arrays respectively. The integers are written in an I9 format and the real numbers in an E12.4 format.

RESTRICTIONS:
GENI writes arrays of integers only. GENR writes arrays of real numbers only.

CALLING SEQUENCE:
GENOUT (A, ISTRT, ISTP, 'NAME')
GENI (A, ISTRT, ISTP, 'NAME')
GENR (A, ISTRT, ISTP, 'NAME')

where A - is the array location
ISTRT - is the first value in A being written
ISTP - is the last value in A being written
'NAME' - is a title of 22 Hollerith words for identification
SUBROUTINE NAME: FLUX

PURPOSE:

Subroutine FLUX permits doublet time variant curve values stored on magnetic tape unit NFLXTP to be read into NCRV arrays starting at array DATA when the mission time exceeds DQTIME. The flux tape must be generated prior to the run using a GE routine LTVFTP. This routine generates the flux tape in the following format:

Record No. 1
First Read Time

Record No. 2
Number of points on first curve (Integer), first curve independent variables, first curve dependent variables, number of points on second curve, second curve independent variables, second curve dependent variables, etc. for all curves.

Record No. 3
Second Read Time

Record No. 4
Same as Record No. 2 except with new values

Record No. 5
Third Read Time

Etc. until all blocks of data are on tape.

Subroutine FLUX writes the values from the appropriate NFLXTP record into the arrays defined by DATA and NCRV in the proper doublet array format. Flux values should be input into the heat flux arrays (DATA1---DATA_NCRV) initially if the user doesn't want the values to be read from the tape at the start of the problem. The value of QTIME should initially be the value of the time the first read is desired.

RESTRICTIONS:
The following restrictions apply:

(1) The initial block of curve data must be input on cards or data
(2) Particular curves must have the same number of points on each block of data read in as were input on cards initially.

(3) Each curve may have a different number of points.

(4) The first point on each curve in each block of data must be the same as the last point on that curve in the previous block of data.

(5) All incident heat curves must be in a single block by themselves.

**CALLING SEQUENCE:**

```
FLUX(NFLXTP, DATA, NCRV, DQTIME, QTIME)
```

where

- **NFLXTP** - logical unit to which the flux tape is assigned. Must be supplied by a user constant.
- **DATA** - starting location (IC) for flux curves
- **NCRV** - number of flux curves to be updated from the flux tape
- **DQTIME** - time scale shift for flux curves. DQTIME is added to each independent value for each flux curve read from NFLXTP.
- **QTIME** - the last point on the latest set of flux curves read from NFLXTP. (QTIME = FLXTIM + DQTIME, where FLXTIM is the time read from the flux tape) must be supplied by user constant.
SUBROUTINE NAME: TIMCHK

PURPOSE:
Subroutine TIMCHK compares the elapsed computer time against the requested computer time, RTIME, and terminates the run if RTIME is exceeded by the elapsed time. If the second argument, KODE, is non-zero an output of computer time used will be printed out on each call to TIMCHK. Thus, a call to TIMCHK in VARIABLES 2 should normally be with KODE=0. If the output of computer time used is desired, TIMCHK should be called from OUTPUT with KODE ≠ 0. The most desirable procedure is to supply two calls to TIMCHK: (1) a call in VARIABLES 2 with KODE = 0 and (2) a call in OUTPUT with KODE ≠ 0.

RESTRICTIONS:
KODE should zero when called from VARIABLES 1 or 2.

CALLING SEQUENCE:
TIMCHK (RTIME, KODE)
where RTIME = maximum computer time requested
KODE = print code: = 0, computer time used is not printed out
  ≠ 0, computer time used is printed out on each call to TIMCHK
SUBROUTINE NAME: REVPOL

PURPOSE: This subroutine performs single variable linear interpolation on a doublet array of X,Y pairs in the same manner as D1DEGL except in reverse order. The array is interpolated in reverse order to obtain the value of independent variable, X, which corresponds to the input dependent variable, Y.

RESTRICTIONS: All values must be floating point numbers.

CALLING SEQUENCE:

REVPOL (Y,A(IC),X)

where Y - input value of dependent variable
A - Doublet array of X,Y pairs
X - output value of independent variable
5.0 SAMPLE PROBLEM

A sample problem was prepared for the SINDA routine to demonstrate the input and output for a typical thermal/flow analysis problem. A schematic of the problem is shown in Figure 10. The problem consists of 8 two dimensional radiator panels, each modeled by two flow paths (one for the main panel of 11 tubes and one for the prime bypass tube). Contained in the system are a pump, a bypass valve (valve No. 1) and a stagnation valve between the two flow paths. The heat load to the radiator system comes through a counter flow heat exchanger which has a controlled inlet temperature of 40°F. The fluid is Freon 21 in the radiator system and water on the cooled side of the heat exchanger. The nodal subdivision for the fluid system is shown in Figure 10. The structure nodal subdivision is shown in Figure 11.

The output for the run is presented in Table 3. A selected few items were plotted using the plot package described in Appendix B. These are presented in Figures 12 thru 17.
FIGURE 10 FLUID MODEL OF THE SAMPLE PROBLEM

xx - Fluid temperature lumps
X - Pressure Nodes
O - Tube numbers
vx - Valve numbers
FIGURE 11 STRUCTURE MODEL FOR THE SAMPLE PROBLEM
TABLE 3
OUTPUT FOR SAMPLE PROBLEMS

| ASG G: A05462 | 25 FEB 73 | 19:19:46 |
| ASG 2: A05793 | 25 FEB 73 | 19:19:46 |
| ASG T: A02293 | 25 FEB 73 | 19:19:46 |
| ASG O, J, K, M | 25 FEB 73 | 19:19:46 |
STARTING ADDRESS 014000
CORE LIMITS 014000 047745 050366 163771 163772 163777

PREPRO/CODE
0 050366-050425
1 014000-014454

NOTES /RALE5
0 050426-050432
1 014455-014547
2 050433-050450

WDDING/RLECS
1 01560-015531
2 050451-050466

FTABS /CODE
0 050502-050665

IFDTS /RALE4
1 015532-016467
2 050451-050666

NBNTS/RLECS
1 016470-016714
2 050466-050759

NOTINS/RLECS
1 016715-017364
2 050755-051020

PACKS/CODE
1 017365-017430

DEPTH /******
0 051021-051026

AFLDS /RALE2
1 017431-017453

MBDVS /RLECS
0 051027-051123
SAMPLE PROBLEM FOR SINDA VERSION 9

**NIKPS/RECS**
0 051214-051216
1 051217-051219
2 051220-051222

**NIKPS/RECN**
1 020517-020727
2 051223-051225

**NIERG/RECS**
0 051303-051305
1 020730-021223
2 051306-051402

**NIERG/RECN**
0 051403-051572
1 021236-021700

**NIJOUT/RECS**
1 021701-022132
2 051573-051575

**NIJUFF/RL23**
1 022133-022155
2 051575-052605

**GROUNDS/RECS**
1 022156-022250

**DUMSUB/CODE**
0 052604-052650
1 022251-022335

**LSTOPS/RECS**
1 022336-022347

**NEWS/COCE**
0 052651-052705
1 022356-022365

**TITLE/COCE**
0 052706-053006
1 022366-022445

**CRDPEL/*****/
0 053007-054023

**TAPE/*****/
0 054024-054035

**BLE9UF/*****/
0 054036-055105

**TABLE/*****/
0 055106-055375

**DATA/*****/
0 055376-055417
SAMPLE PROBLEM FOR SIANDA VERSION 7

LOGIC /*****
  0 05420-05519

FLAG/*****
  0 055545-055526

SRDCOM/*****
  0 055527-056130

BUCKET/*****
  0 056131-156500

POINT /*****
  0 156501-156575

CHECK/*****
  0 156576-156747

FLAGS /*****
  0 156750-156752

JPS /*****
  0 156753-154753

CIAGE/*****
  0 156754-157333

FINIPS/PLECS
  0 022446-022707
  2 157232-157334

SEARCH/ CODE
  0 157335-157351
  3 022710-022774

MATD3/CODE
  3 022775-023433

BLKRD/CODE
  0 157352-157630
  3 023434-024327

WATBLX/CODE
  1 024330-024341

STFFB / CODE
  0 157831-157833
  1 024724-024723

FINCAD/CODE
  0 157844-157817
  1 024724-025045

SQUEEZ/CODE
  0 157720-157722
  1 024724-025045
SAMPLE PROBLEM FOR SINDA VERSION 9

SREAD/CODE
0 157749-160035
1 026016-029350

SREAD/CODE
0 160036-160055
1 029351-029999

RDILK /CODE
0 160056-160070
1 029999-029999

PERS /CODE
0 160071-160132
1 029999-029999

MITAN /CODE
0 160133-160187
1 029999-029999

EUROMC /CODE
0 160188-160249
1 029999-029999

**EDIT** /CODE
0 160145-161056
1 026005-027374

**ADDU** /CODE
0 160145-160306
1 026005-027063

BUFFAP /CODE
0 160305-160317
1 027064-027223

BUFFCOF /CODE
0 160320-160324
1 027224-027306

TOCENT /CODE
0 160327-160350
1 027307-028753

FINDOR /CODE
0 160351-160357
1 027535-027613

PPCH /CODE
0 160365-160372
1 027614-027641

AZILUV /CODE
0 160373-160454
1 027642-027746

TDDOF /CODE
0 160455-160462

99
SAMPLE PROBLEM FOR SINDA VERSION 9

TCCWRIT/COCE
0 160463-160571
1 030036-030151

ROSET /CODE
0 160572-160601
1 030152-030223

MURTIT/COCE
0 160602-160896
1 030224-030310

EDITIN/COCE
0 160897-160963
1 030311-030430

RGGG01/COCE
0 160964-160975

ETOOG /COCE
1 030431-030465

CLOCK /COCE
0 160976-160996
1 030546-030546

NEWMOD/COCE
0 161000-161010
1 030947-030947

UPDATE/COCE
0 161011-161110
1 032606-032606

UPDATE/COCE
0 161111-161117
1 032607-033613

PAWITE/COCE
0 161220-161275
1 033414-034043

COPPAS/COCE
0 161276-161415
1 034044-034275

SINDA/COCE
0 161416-161664
1 034276-034533

GREADI/COCE
0 161665-161743
1 034534-035074

STATW/COCE
0 161744-162125

100
SAMPLE PROBLEM FOR SIMOA VERSION 9

SETFLD/icode
0 162216-162245
1 035614-035779

MOVE /code
0 162246-162269
1 035775-036127

NCOL /code
0 162265-162277
1 036130-036150

AYM /code
0 162272-162302
1 036151-036212

*BUFFALO////////
0 162303-163276

*JOINNEW////////
0 163277-163373

*PUSCAD////////
0 163374-163459

**GENLINK/icode
0 160145-160267
1 026005-026316

**PSEUDO/icode
0 160268-160297
1 026005-030131

**PCSZ /code
0 160272-160301
1 030132-030177

**CODERD/ICODE
0 160145-160226
1 026005-030640

IMPRE /CODE
0 160427-160520
1 030641-033113

**DATARD/icode
0 160521-161417
1 031110-035727

**RATES/ICODE
0 161455-162757
1 035739-036411

**CONVAR/ICODE
0 162745-162915
1 036612-036737

DATE 250273 PAGE 4

101
SAMPLE PROBLEM FOR SINDA VERSION 9

**TPRM**

**DATA**

**RELAY**

**WRTDATA**

**WRTMAT**

**GSMAP**

**INCON**

**SETFMT**

**GERM**

**NOD**

**CNSD**

**PRES**

**SIND**

**SETPAR**

**LPRINT**
SAMPLE PROBLEM FOR SINDA VERSION 9

* SPLIT /CODE
  0  161227-161237
  1  031210-031712

* SKIP /CODE
  0  160145-160170
  1  026605-026770

END OF ALLOCATION 1103 0039A 09099
A description of the modifications to each version of SINDA is contained on comment cards in subroutine NEWS.
SAMPLE PROBLEM FOR SINDA VERSION 9

BEGIN SINDA PREPROCESSOR, VERSION 8, AT 14:15:03
SAMPLE PROBLEM FOR SINDA VERSION 9

BCD THERMAL LPCS

BCD 6 SAMPLE PROBLEM FOR SINDA VERSION 9

END

BCD NODE DATA

REM NODE,NUM,INC, TI, ACT, CONST, FFLUID LPCS

<table>
<thead>
<tr>
<th>NODE</th>
<th>NUM</th>
<th>INC</th>
<th>TI</th>
<th>ACT</th>
<th>CONST</th>
<th>FFLUID LPCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>501</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.012096</td>
<td>FLUID LPCS</td>
</tr>
<tr>
<td>502</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>506</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>507</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>508</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>509</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>510</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>511</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>513</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>.00305</td>
<td></td>
</tr>
</tbody>
</table>

SINCA VERSION 9

CO FLUID TUBE LPCS

<table>
<thead>
<tr>
<th>NODE</th>
<th>NUM</th>
<th>INC</th>
<th>TI</th>
<th>ACT</th>
<th>CONST</th>
<th>FFLUID TUBE LPCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>501</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>502</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>506</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>507</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>508</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>509</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>510</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>511</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>513</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td>0</td>
<td>6</td>
<td>70</td>
<td>0</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 12

END

RELATIVE NODE NUMBERS

ACTUAL NODE NUMBERS

NODE ANALYSIS... DIFFUSION = 234, ARITHMETIC = 5, BOUNDARY = 4, TOTAL = 238

BCD SOURCE DATA

REM NODE, ALTIRE, CONST $
<table>
<thead>
<tr>
<th>Site</th>
<th>#</th>
<th>A15</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>212</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>213</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>216</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>217</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>221</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>222</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>225</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>226</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>228</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>229</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>231</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>233</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>237</td>
<td>A15</td>
<td>16.270 s</td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>251</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>252</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>253</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>254</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>255</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>257</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>258</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>259</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>260</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>261</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>263</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>264</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>265</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>266</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>267</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>268</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>269</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>270</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>271</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>272</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>273</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>274</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>275</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>276</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>277</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>278</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>279</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>280</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>281</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>282</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>283</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>284</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>285</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>286</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>287</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>288</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>289</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>290</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>291</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>292</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>293</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>294</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
<tr>
<td>295</td>
<td>A15</td>
<td>0.431 s</td>
<td></td>
</tr>
</tbody>
</table>
Flow conductors identified in array 17, page 114

* Comments added to listing of input data to clarify input of flow systems
SAMPLE PROBLEM FOR SINDA VERSION 9

| GEN | 201, 117, | 1, 1, 1, 201, | 1. 100. | | REM | NO | NOG | 16 | NA | ING | NB | INB | 6 |
|-----|------------|--------------|----------|----------------------------------|-----------------|-----------------|-----------------|-----------------|
| GEN | 401, 8, 1, 202, 6, 400, 0, 2.59E-8 | | | | GEN | 409, 8, 1, 203, 6, 400, 0, 2.59E-8 | | | | GEN | 417, 8, 1, 204, 6, 400, 0, 2.59E-8 | | | | GEN | 435, 8, 1, 250, 6, 400, 0, 0.66E-9 | | | | GEN | 491, 8, 1, 251, 6, 400, 0, 0.66E-9 | | | | GEN | 499, 8, 1, 252, 6, 400, 0, 0.66E-9 | | | | GEN | 457, 8, 1, 253, 6, 400, 0, 0.66E-9 |

<table>
<thead>
<tr>
<th>RELATIVE CONDUCTOR NUMBERS</th>
<th>1</th>
<th>11</th>
<th>21</th>
<th>31</th>
<th>41</th>
<th>51</th>
<th>61</th>
<th>71</th>
<th>81</th>
<th>91</th>
<th>101</th>
<th>111</th>
<th>121</th>
<th>131</th>
<th>141</th>
<th>151</th>
<th>161</th>
<th>171</th>
<th>181</th>
<th>191</th>
<th>201</th>
<th>211</th>
<th>221</th>
<th>231</th>
<th>241</th>
<th>251</th>
<th>261</th>
<th>271</th>
<th>281</th>
<th>291</th>
<th>301</th>
<th>311</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTUAL CONDUCTOR NUMBERS</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>100</td>
<td>110</td>
<td>120</td>
<td>130</td>
<td>140</td>
<td>150</td>
<td>160</td>
<td>170</td>
<td>180</td>
<td>190</td>
<td>200</td>
<td>210</td>
<td>220</td>
<td>230</td>
<td>240</td>
<td>250</td>
<td>260</td>
<td>270</td>
<td>280</td>
<td>290</td>
<td>300</td>
<td>310</td>
<td>312</td>
</tr>
</tbody>
</table>

| CONNECTION | - | Connection conductor identified in array 16, page 112 |
| RADIATION | - | - |

DATE 250273 PAGE 15

110
**BCD CONDUCTOR ANALYSIS...**

**LINEAR = 248, RADIATION = 69, TOTAL = 312, CONNECTIONS = 312**

---

**BCD CONSTANT DATA**

<table>
<thead>
<tr>
<th>TIME</th>
<th>k</th>
<th>DIPPE</th>
<th>RLODP</th>
<th>RALIC</th>
<th>OALIC</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td></td>
<td>0.01</td>
<td>100</td>
<td>0.01</td>
<td>0.01</td>
<td>1.0</td>
</tr>
</tbody>
</table>

---

**BCD ARRAY DATA**

1. **FREON-21 SPECIFIC HEAT**

| -400 | 110. | -218. | .223 | 217. | .223 | 3.723 |
| -212 | .223 | 217.  | 220. | 231  | 0    | .237  |
| 40   | .254 | 190.  | .280 | 150. | 81   | .295  |

2. **FREON-21 DENSITY**

| -400 | 110. | -218. | 110. |
| -212 | 110. | -214. | 110. |
| 40   | 99.25| -60.  | 96.  |

3. **FREON-21 DENSITY TIMES SPECIFIC HEAT**

| -400 | 24.53 | -211. | 24.53 | .217  | .217  | 4.0993 |
| -212 | .217  | 24.53 | -160. | 23.30 |
| 40   | 21.59 | 90.   | 21.39 |

4. **ALUMINUM SPECIFIC HEAT**

| -400 | 100. | .175  | 100. |
| -200 | .192 | 100.  | .204 |
| 200  | .214 |

5. **FREON-21 VISCOSITY**

| -400 | 19.1  | -212. | 19.1  |
| -200 | 16.55 | -201. | 16.55 |
| 100  | 25.0  | -197. | 25.0  |

6. **FREON-21 THERMAL CONDUCTIVITY**

| -400 | 6.70  | 100.  | .726  | 160.  | .561  |
| 200  | .396  |       |       |       |       |
ADAT array for convection conductors; described on page 49 and the last argument to subroutine CONV1 on page 121.
<table>
<thead>
<tr>
<th>Sample Problem for SINDA Version 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>*<em>G243, 1.35 22.</em> T43, 1, 12.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G244, 1.17 22.</em> T44, 2, 18.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G245, 1.17 22.</em> T45, 2, 15.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G246, 1.17 22.</em> T46, 2, 18.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G247, 1.17 22.</em> T47, 2, 12.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G248, 1.35 22.</em> T48, 1, 12.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G249, 1.35 16.</em> T49, 1, 12.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G250, 0.082,16.</em> T50, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G251, 0.082,16.</em> T51, 4, 12.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G252, 0.082,16.</em> T52, 4, 12.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G253, 0.082,16.</em> T53, 4, 13.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G254, 1.35 16.</em> T54, 1, 25.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G255, 0.082,15.</em> T55, 3, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G256, 0.082,15.</em> T56, 3, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G257, 0.082,15.</em> T57, 4, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G258, 0.082,15.</em> T58, 4, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G259, 0.082,15.</em> T59, 9, 6.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G260, 0.082,15.</em> T60, 3, 11.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G261, 0.082,15.</em> T61, 1, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G262, 0.082,13.</em> T62, 4, 5.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G263, 0.082,13.</em> T63, 4, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G264, 0.082,13.</em> T64, 4, 5.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G265, 0.082,13.</em> T65, 4, 6.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G266, 0.082,13.</em> T66, 3, 11.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G267, 1.35 14.</em> T67, 1, 12.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G268, 0.082,14.</em> T68, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G269, 0.082,14.</em> T69, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G270, 0.082,14.</em> T70, 4, 13.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G271, 0.082,14.</em> T71, 4, 13.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G272, 1.35 13,</em> T72, 1, 25.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G273, 1.35 12.</em> T73, 1, 25.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G274, 0.082,30.</em> T74, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G275, 0.082,30.</em> T75, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G276, 0.082,30.</em> T76, 4, 12.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G277, 1.35 30.</em> T77, 1, 25.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G278, 0.5625,29.</em> T78, 3, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G279, 0.082,29.</em> T79, 4, 5.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G280, 0.082,29.</em> T80, 4, 5.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G281, 0.002,29.</em> T81, 4, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G282, 0.002,29.</em> T82, 4, 5.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G283, 0.002,29.</em> T83, 4, 6.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G284, 0.5625,29.</em> T84, 3, 11.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G285, 0.5625,31.</em> T85, 3, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G286, 0.002,31.</em> T86, 4, 5.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G287, 0.002,31.</em> T87, 4, 5.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G288, 0.002,31.</em> T88, 4, 5.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G289, 0.002,31.</em> T89, 4, 6.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G290, 0.5625,31.</em> T90, 3, 11.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G291, 1.35 32.</em> T91, 1, 12.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G292, 0.082,32.</em> T92, 4, 12.25 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G293, 0.082,32.</em> T93, 4, 12.5 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G294, 0.082,32.</em> T94, 4, 12.75 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G295, 0.082,32.</em> T95, 4, 13.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G296, 1.35 32.</em> T96, 1, 25.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G297, 0.5625,1.</em> T97, 3, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G298, 0.5625,2.</em> T98, 3, 5.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G299, 2.25 3.</em> T99, 6, 20.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G300, 7.875, 4.</em> T100, 8, 7.0 , 1.0, 1.0</td>
</tr>
<tr>
<td>*<em>G301, 7.875, 9.</em> T101, 8, 7.0 , 1.0, 1.0</td>
</tr>
</tbody>
</table>

ADAT array (Cont'd)
SAMPLE PROBLEM FOR SINDA VERSION 9

`G02, .281, .10, T102, 6, 2.9, 1.0, 1.0
G03, .281, .14, T103, 6, 2.5, 1.0, 1.0
G04, .7075, 20, T104, 8, 7.0, 1.0, 1.0
G05, .7075, 25, T105, 8, 7.0, 1.0, 1.0
G06, 2.25, 26, T106, 5, 20.0, 1.0, 1.0
G07, 2.25, 71, T107, 5, 20.0, 1.0, 1.0
G08, .7075, 12, T108, 8, 7.0, 1.0, 1.0
G09, .7075, 17, T109, 8, 7.0, 1.0, 1.0
G10, .281, .18, T110, 6, 2.5, 1.0, 1.0
G11, .281, .27, T111, 6, 2.5, 1.0, 1.0
G12, .281, .28, T112, 8, 7.0, 1.0, 1.0
G13, .281, .33, T113, 8, 7.0, 1.0, 1.0
G14, 2.25, 26, T114, 5, 20.0, 1.0, 1.0
G15, 5.62, 25, T115, 7, 50.0, 1.0, 1.0
G16, .225, .36, T116, 9, 2.0, 1.0, 1.0
G17, .5625, 37, T117, 3, 5.0, 1.0, 1.0

END

FLOW CONDUCTOR DATA

G1, T11, 6 c
G2, T12, 6 c
G3, T13, 6 c
G4, T14, 6 c
G5, T15, 6 c
G6, T16, 6 c
G7, T17, 6 c
G8, T18, 6 c
G9, T19, 6 c
G10, T20, 6 c
G11, T21, 6 c
G12, T22, 6 c
G13, T23, 6 c
G14, T24, 6 c
G15, T25, 6 c
G16, T26, 6 c
G17, T27, 6 c
G18, T28, 6 c
G19, T29, 6 c
G20, T30, 6 c
G21, T31, 23 c
G22, T32, 23 c
G23, T33, 23 c
G24, T34, 23 c
G25, T35, 23 c
G26, T36, 23 c
G27, T37, 23 c
G28, T38, 23 c
G29, T39, 23 c
G30, T40, 23 c
G31, T41, 23 c
G32, T42, 23 c
G33, T43, 23 c
G34, T44, 23 c
G35, T45, 23 c
G36, T46, 23 c
G37, T47, 23 c
G38, T48, 23 c
G39, T49, 23 c
G40, T50, 23 c
G41, T51, 23 c
G42, T52, 23 c
G43, T53, 23 c
G44, T54, 23 c
G45, T55, 23 c
G46, T56, 23 c
G47, T57, 23 c
G48, T58, 23 c
G49, T59, 23 c
G50, T60, 23 c
G51, T61, 23 c
G52, T62, 23 c
G53, T63, 23 c
G54, T64, 23 c
G55, T65, 23 c
G56, T66, 23 c
G57, T67, 23 c
G58, T68, 23 c
G59, T69, 23 c
G60, T70, 23 c
G61, T71, 23 c
G62, T72, 23 c
G63, T73, 23 c
G64, T74, 23 c
G65, T75, 23 c
G66, T76, 23 c
G67, T77, 23 c
G68, T78, 23 c
G69, T79, 23 c
G70, T80, 23 c
G71, T81, 23 c
G72, T82, 23 c
G73, T83, 23 c
G74, T84, 23 c
G75, T85, 23 c
G76, T86, 23 c
G77, T87, 23 c
G78, T88, 23 c
G79, T89, 23 c
G80, T90, 23 c
G81, T91, 23 c
G82, T92, 23 c
G83, T93, 23 c
G84, T94, 23 c
G85, T95, 23 c
G86, T96, 23 c
G87, T97, 23 c
G88, T98, 23 c
G89, T99, 23 c
G90, T100, 23 c
G91, T101, 23 c
G92, T102, 23 c
G93, T103, 23 c
G94, T104, 23 c
G95, T105, 23 c
G96, T106, 23 c
G97, T107, 23 c
G98, T108, 23 c
G99, T109, 23 c
G100, T110, 23 c
G101, T111, 23 c
G102, T112, 23 c
G103, T113, 23 c
G104, T114, 23 c
G105, T115, 23 c
G106, T116, 23 c
G107, T117, 23 c

ADAT array (cont'd)

ADAT array described on page 53 and the third argument to FLOCN1 on page 121
ADAT1 array (cont'd)
SAMPLE PROBLEM FOR SINDA VERSION 9

ADAT1 array (cont'd)

AFLOW array described on page 71 and the first argument in the call to PFCS on page 121 (also first argument to CONV1)

AW array described on page 72 and referenced in array 20 above

APN array described on page 72

AGF array described on page 72
SAMPLE PROBLEM FOR SINDA VERSION 9

AVP array described on page 72

AIFR array described on page 72

AFT array containing the fluid type data described on page 72 and referenced in array 51.

AFR array described on page 72

APD array described on page 73

ADAT array described on page 71 and the second argument in the call to PFCS on page 121

APR array described on page 73 and identified in array 30

ASOL array described on page 74 and identified in array 30

ANET array for the main network described on page 74 and identified in array 30.
SAMPLE PROBLEM FOR SINDA VERSION 9

---

END

---

AP array containing pump data described on page 77 and identified in array 30 described on page 76 and referenced in array 30(AVLS) and array 33(AVL)

ADi arrays for the main network; described on page 75 and referenced in array 33

AVLV arrays containing valve data described on page 76 and referenced in the AVL array 36

ANE network for subnetwork No. 1; described on page 74 and identified in ANET for the main network, array 33

ADi arrays for subnetwork No. 1 described on page 75 and referenced in array 40
SAMPLE PROBLEM FOR SINDA VERSION 9

55, T119, 1, T218, END

56, T230, 2, T240, END

57, T330, 2, T340, END

58, T430, 2, T440, END

59, T530, 2, T540, END

60, T630, 2, T640, END

ADi arrays for subnetwork No. 1 (Cont'd)

ANET array for subnetwork 2; described on page and identified in ANET for the main network, array 33
AD1 array for subnetwork No. 2 described on page 75 and referenced in array 43.
SAMPLE PROBLEM FOR SIMCA VERSION 9

DATE 250273 PAGE 26

BCD 9SHUTTLE ECS RADIATOR FLOW SYSTEM
END

BCD 9FLOW RATES (LB/HR)
END

BCD 9PRESSURES (LB/FT**2)
END

BCD 9VALVE POSITIONS
END

BCD 9PRESSURE DROPS (LB/FT**2)
END

END

ARRAY ANALYSIS... NUMBER OF ARRAYS = 75 TOTAL LENGTH = 2511

BCD EXECUTION

DIMENSION T(2000)
NOIM = 2000
NIN = 0

PFCS(A20,A30,A100+1)
CNBACK

BCD 3VARIABLES 1
DIDEG(TIPEN,A11,T198)
HEFF (0.9,500.,A21+37, 1.0, A1, T198, T117, T199, T200)
FLOCK(A21,A1,A7)
CONV(A20,A31,A16)

BCD 3VARIABLES 2
PFCS(A20,A30,A100+1)
TIMCHK(K,0)
KSTRY(A22,A21,A11,DTIMEU)

BCD 3OUTPUT CALLS

TPRINT
FLPANT(A21,A119+1)
FLPANT(A20,A113+1)
FLPANT(A22,A111+1)
FLPANT(A24,A112+1)
TIMCHK(K,1)

END

121.
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 27
25 FEB 73

0 10T CUR
1. ENS
2. ING
3. TRG
4. ENS
5. ING
6. TRG
7. ING
8. RTR
9. RTR
10. RTR
11. RTR
12. RTR
13. RTR
14. RTR
15. RTR
16. RTR
17. RTR
18. RTR
19. RTR
20. RTR
21. RTR
22. RTR
23. RTR

END OF FILE -- UNIT 6
END CUR LCC 1102-0039 L9

122
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 28
25 FEB 73 14:16:7

MAIN PROGRAM

STORAGE USED: CODE(19) 000014; DATA(9) 000001; BLANK COMMON(2) 000000

COMMON BLOCKS:

0003 TITLE 000024
0004 TEXP 000036
0005 CAP 000032
0006 SOURCE 000032
0007 COND 000040
0010 PCI 000055
0011 PC2 000052
0012 RCONST 000020
0013 ARRAY 000017
0014 FIXCON 000062
0015 DNPENS 000011
0016 LOGIC 000004

EXTERNAL REFERENCES (BLOCK, NAME)

0017 INPUT
0020 EXECIN
0021 NSTOP2

STORAGE ASSIGNMENT (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0001 000005 IL 0013 000000 A 0014 000022 ARLICA 0014 000035 ARLICE 0014 000012 A TPFFA
0014 000007 ATPFFA 0014 000013 BACKUP 0014 000040 BALENC 0014 000020 C 0014 000001 CSFAC
0014 000206 CSFAC 0014 000020 CSFAC 0014 000027 CSPCAL 0014 000030 CSPCAL 0014 000014 DAPA
0014 000011 DAPA 0014 000031 DAPA 0014 000032 DAPA 0014 000057 DIPFEN 0014 000026 DIPFEN
0014 000036 DIPFEN 0014 000030 DIPFEN 0014 000035 DIPFEN 0014 000031 DIPFEN
0007 000000 G 0003 000000 H 0014 000046 I TEST 0014 000047 I TEST 0014 000050 K
0014 000050 K TEST 0016 1 000003 LARRAY 0014 000040 LARRAY 0014 000030 LARRAY
0015 000100 LENA 0014 000003 LDISPLAY 0014 000030 LDISPLAY 0014 000023 LDISPLAY
0015 000007 LSS1 0015 000007 LSS2 0014 000030 LTEST 0014 000032 LTEST 0014 000031 LTEST
0015 000006 LTEST 0014 000046 LTPFAC 0014 000042 LTPFAC 0014 000030 LTPFAC 0014 000035 LTPFAC
0015 000003 LTPFAC 0014 000046 LTPFAC 0015 000030 LTPFAC 0014 000023 LTPFAC
0014 000036 LTEST 0014 000056 LTEST 0014 000030 LTEST 0014 000035 LTEST
0014 000055 LTEST 0014 000056 LTEST 0014 000030 LTEST 0014 000035 LTEST

GS01 1+ COMMON #FILE$ N
GS03 2+ COMMON #STATE$ T
GS04 3+ COMMON #STATE$ C
GS05 4+ COMMON #STATE$ Q
GS10 5+ COMMON #STATE$ S
GS17 6+ COMMON #STATE$ SE2
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 29

00110 1* COMMON /PC2/ SEQ2
00111 8* COMMON /PCST/ K
00112 9* COMMON /ARRAY/ A
00113 10* COMMON /FIXCON/ TIP0, DTIP0, TIP1, CSGlobal,
00113 11* IRLC0P, TIMP0C, OPE1T, OPE1T, DAMPA,
00113 12* IDAMP0C, BTMPC, BACKUP, TIP0, TIP1,
00113 13* IOTMP0C, ATMP0C, CSGlobal, OUTPUT, WiLC0C,
00113 14* IC0PPC', DTMP0C, DTMP0C, CSGlobal, CSGlobal,
00113 15* ICSGACL, OALY1C, OALY1C, LINECT, PASECT,
00113 16* IDAMP1C, AUTMPC, MARLKC, NAMTPC, IVE1ST,
00113 17* IJTEST, RTEST, LTEST, RTEST, RTEST,
00113 18* IISTEST, TIEST, UIEST, UIEST, UIEST,
00113 19* ITEST, TTEST, UTEST, VTEST, LAFAC,
00113 20* ICENTAL
00114 21* COMMON /DIMENS/ NHA, NHA, NHA, NCT, NCT, LSO1, LSO2, LENA
00115 22* DIMENSION H(20)
00116 23* COMMON /LOGIC/ LND0E, LC0ND, LCONST, LARRAY
00117 24* LOGICAL LND0E, LC0ND, LCONST, LARRAY
00118 25* DIMENSION T(238), C(234), O(239), G(312), K(131), A(251),
00118 26* ISEQ1(629), SEQ2(298)
00119 27* LND0E = .TRUE.
00120 28* LC0ND = .TRUE.
00121 29* LCONST = .TRUE.
00122 30* LARRAY = .TRUE.
00123 31* I CALL INPUT
00124 32* CALL EXECUT
00125 33* GO TO 1
00126 34* END

END OF COMPILATION: NO DIAGNOSTICS.
SUBROUTINE EXECPN  ENTRY POINT 00021

STORAGE USED: CODE(1) 000023; DATA(0) 000005; BLANK COMMON(2) 000000

COMMON BLOCKS:

0003 TITLE 000024
0004 TEMP 000001
0005 CAP 000001
0006 SOURCE 000001
0007 CONS 000001
0010 PC1 000001
0011 PC2 000001
0012 KNST 000001
0013 ARRAY 000001
0014 FIXCON 000042
0015 DIPENS 000011
0016 YSPACE 003722

EXTERNAL REFERENCES (BLOCK, NAME)

0017 FFCES
0020 CNBACK
0021 .KARR35

STORAGE ASSIGNMENT (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0013 A 000000 A 0114 000022 ARYXCA 0014 000035 ARYXCC 0014 000012 ATMPAC 0014 000017 ATMPCC
0014 000013 BACKUP 0014 000040 KALENG 0015 000000 C 0014 000003 CSFPAC 0014 000026 CSFPAC
0014 000020 CSGRAL 0014 000027 CSGRAL 0015 000039 CSGRAL 0014 000018 CARRPA 0014 000011 CARRPA
0014 000031 DCPRAM 0014 000032 DCPRA 0014 000037 DTIPAC 0014 000021 DTIPAC 0014 000024 DTIPAC
0014 000001 DTPAC 0014 000005 DTPAC 0014 000014 DTPAC 0014 000037 DTPAC 0015 000000 C
0015 000050 KTEST 0015 000000 KTEST 0015 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST
0014 000050 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST 0014 000000 KTEST

0016 1* SUBROUTINE EXECPN
0017 2* COMMON /TITLE/ T
0018 3* COMMON /TYPE/ T
0019 4* COMMON /CAP/ C
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 31

GO104  5= COMMON /SOURCES/ G
GO107  6= COMMON /SEND/ G
GO110  7= COMMON /PC1/ SEQ1
GO111  8= COMMON /PC2/ SEQ2
GO112  9= COMMON /SEND/ R
GO113 10= COMMON /ARRAY/ A
GO114 11= COMMON /FIXONS/ TEMP, DTIMEU, TIMEND, CSGFAC
GO117 12= INLOOP, DTPCCA, SPEITR, DTIMR, CAOMA
GO118 13= IDAMPO, ATFRCA, BACKUP, TPFEO, TIPEM
GO119 14= IDTPCC, ATFRCC, CSGMIN, OUTPUT, ARLICA
GO120 15= IDOPCI, DTIMEL, DTIMR, CSGNAT, CSGRAL
GO121 16= CSGRCL, DFRICA, DFRICR, LNETC, PAGERT
GO122 17= JARLCC, LSPCC, ENGBAL, BALENC, HNOTFY
GO123 18= INDEXM, HTPAC, HAMILE, HATMP, ITEST
GO124 19= IJTEST, ITEST, ITEST, MTEST, MTEST, RTTEST
GO125 20= IJTEST, ITEST, ITEST, VTEST, VTEST, LAFAC
GO126 21= IDENTR
GO127 22= COMMON /DIMENS/ K, HA, KN, HGT, HCT, NAT, LS100, LS02, LENR
GO128 23= DIMENSION K(20)
GO129 24= COMMON /XSPACE/ ACIM, NTH, 1
GO130 25= DIMENSION T(1), C(1), O(1), G(1), K(1), A(1)
GO131 26= DIMENSION XX(1), YY(1)
GO132 27= EQUIVALENCE (K, IK), (1, NI)
GO133 28= DIMENSION XI(200)
GO134 29= NTH = 0
GO135 30= NTH = 0
GO136 31= CALL PCS (RI557), RI5NE, RI2463)
GO137 32= CALL CNECK
GO138 33= RETURN
GO139 34= END

END OF COMPILATION: NO DIAGNOSTICS.
SUBROUTINE VARBL1  ENTRY POINT 000041

STORAGE USED:  CODE(1) 000043,  DATA(0) 000010,  BLANK COMMON(2) 000000

COMMON BLOCKS:

0003 TITLE 000024
0004 TEMP 000001
0005 CAP 000001
0006 SOURCE 000001
0007 COMMON 000001
0010 PCI 000001
0011 PCI 000001
0012 KONST 000001
0013 ARRAY 000001
0014 FIXCON 000002
0015 BIPENS 000011
0016 BSPACE 000003

EXTERNAL REFERENCES  (BLOCK, NAME)

0017 BIDECI
0020 HIEFF
0021 FLOCI
0022 CONVI
0023 RERRG

STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0025 M 000000 A 0014 000022 ARLYCA 0014 000035 ARLYCC 0014 000017 ARLPC
0014 000023 ARLPC 0014 000024 ARLPC 0014 000025 ARLPC 0014 000026 ARLPC
0014 000027 ARLPC 0014 000028 ARLPC 0014 000029 ARLPC 0014 000030 ARLPC
0014 000031 ARLPC 0014 000032 ARLPC 0014 000033 ARLPC 0014 000034 ARLPC
0014 000035 ARLPC 0014 000036 ARLPC 0014 000037 ARLPC 0014 000038 ARLPC
0014 000039 ARLPC 0014 000040 ARLPC 0014 000041 ARLPC 0014 000042 ARLPC
0014 000043 ARLPC 0014 000044 ARLPC 0014 000045 ARLPC 0014 000046 ARLPC
0014 000047 ARLPC 0014 000048 ARLPC 0014 000049 ARLPC 0014 000050 ARLPC
0014 000051 ARLPC 0014 000052 ARLPC 0014 000053 ARLPC 0014 000054 ARLPC
0014 000055 ARLPC 0014 000056 ARLPC 0014 000057 ARLPC

0015 1* SUBROUTINE VARBL1
0013 2* COMMON / TITLE/ w
SAMPLE PROBLEM FOR SINDA VERSION 9  

DATE 250273 PAGE 33

COMMON /STEMP/ T
COMMON /CAP/ C
COMMON /SOURCE/ G
COMMON /COND/ G
COMMON /PRI/ SEQ
COMMON /PC2/ SEQ
COMMON /KINST/ K
COMMON /ARRAY/ A

COMMON /TIME/, TIMEN, DTIMEI, TIMEND, ESFAC,
1ALOOP, DTIPCA, CPEITA, DTIPE, BANPA
1ADPER, ATIPCA, BACKUP, TIPED, TIMEI
1BIPPCA, ATIPPC, CSCHN, OUTPUT, ARLYCA
1CIPPCA, ATIPPC, CSCHN, OUTPUT, ARLYCA
1CISCOCL, ORLYCA, ORLYCC, LINECT, RATEC,
1DARLIC, LSPEC, ENGBL, BALENG, NOCOPY,
1ESEM, NDTPCC, RAARIC, NASPPC, TTEST
1IJUST, KTEST, LTEST, MTST, NTEST
1TTEST, NTEST, UTTEST, VTEST, LABFX,

COMMON /DIMEN/ KND, KNA, KNT, MCTIN, MCT, NET, LS01, LS02, LENA

DIMENSION H(20)
DIMENSION NDIM, NTH, I
DIMENSION T(1), C(1), G(1), K(1), A(I)
DIMENSION T(1), A(I), T(I)
EQUIVALENCE (K, YK), (I, NT)

CALL DIEGIC(TIPEM, A(103), T(235))
CALL XEFF (0.9, 500., (10, 83).10, A1), T(235), T(237)
CALL FLOCNIC (A(153), A(153), A(153))
CALL CONV1 (A(153), A(153), A(214))
RETURN
END

END OF COMPILATION: NO DIAGNOSTICS.
**SUBROUTINE VARBL2**  ENTRY POINT 000026

**STORAGE USED:** CODE1) 000030; DATA0) 000006; BLANK COMMON2) 000000

**COMMON BLOCKS:**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0003</td>
<td>TITLE</td>
<td>000024</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0004</td>
<td>TEMP</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0005</td>
<td>CAP</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0006</td>
<td>SOURCE</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0007</td>
<td>COND</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td>PCI</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PCI2</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td>000001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td>000002</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td>000011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td>000003</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**EXTERNAL REFERENCES** (BLOCK, NAME)

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0017</td>
<td>PFC5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>TPCHK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0021</td>
<td>NICRY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0022</td>
<td>NERR3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STORAGE ASSIGNMENT** (BLOCK, TYPE, RELATIVE LOCATION, NAME)

<table>
<thead>
<tr>
<th>Block</th>
<th>Type</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0013</td>
<td>R</td>
<td>000000 A</td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>C</td>
<td>000000 B</td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>C</td>
<td>000000 C</td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>C</td>
<td>000000 D</td>
<td></td>
</tr>
<tr>
<td>0017</td>
<td>C</td>
<td>000000 E</td>
<td></td>
</tr>
<tr>
<td>0018</td>
<td>C</td>
<td>000000 F</td>
<td></td>
</tr>
<tr>
<td>0019</td>
<td>C</td>
<td>000000 G</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>C</td>
<td>000000 H</td>
<td></td>
</tr>
<tr>
<td>0021</td>
<td>C</td>
<td>000000 I</td>
<td></td>
</tr>
<tr>
<td>0022</td>
<td>C</td>
<td>000000 J</td>
<td></td>
</tr>
</tbody>
</table>

**COMMON**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010</td>
<td>PC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**REFERENCES**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0013</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0017</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0018</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0019</td>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0021</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0022</td>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010</td>
<td>PC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**COMMON**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010</td>
<td>PC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**REFERENCES**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010</td>
<td>PC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**COMMON**

<table>
<thead>
<tr>
<th>Block</th>
<th>Name</th>
<th>Type</th>
<th>Location</th>
<th>Relative Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010</td>
<td>PC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>PC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>KONST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>ARRAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>FIXPOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>DIMENS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0016</td>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 35

00105 4* COMMON /CAP/ C
00106 5* COMMON /SOURCE/ Q
00107 6* COMMON /COND/ G
00110 7* COMMON /PCI/ SER1
00111 8* COMMON /PCI/ SER2
00112 9* COMMON /CONST/ K
00113 10* COMMON /ARRAY/ A
00114 11* COMMON /FIXCOM/ TIMEN, DTIMEN, TIPOSE, CSGFAC,
00114 12* 1DAMP, DTMPC, OREITA, DTIMEN, DAMP,
00114 13* IDAMP, DTMPC, BACKUP, TIMED, TIMEN,
00114 14* IDTMPC, DTMPC, CSGMIN, OUTPUT, ARLICA,
00114 15* ICOP, DTMPC, OREITA, CSGMAX, CSGRAL,
00114 16* ICSGAC, ORLICA, ORLACC, LINFCT, PACECT,
00114 17* IARLCC, LSPEC, ENGBAL, BALENG, NOCOPY,
00114 18* INLorem, NDTMPC, NARLICA, NAMTPC, NTEST,
00114 19* IJTEST, KTTEST, LTTEST, MTTEST, NTTEST,
00114 20* IJTEST, TTTEST, UTTEST, VTEST, LARFAC,
00114 21* INTERNAL
00115 22* COMMON /DIMENS/ ANO, ANG, ANT, NGT, NCT, LAT, LS01, LS02, LENA
00116 23* DIMENSION H(20)
00117 24* COMMON /XSPACE/ R01N, RMT, X
00120 25* DIMENSION X(1), Y(1), Z(1), R(1), A(1)
00121 26* DIMENSION XK(1), RY(1), ZK(1)
00122 27* EQUIVALENCE (X, XK), (Y, YK)
00123 28* CALL PFCS (A(1537), A(1819), A(24631))
00124 29* CALL TIMCHK(K(1), 0)
00125 30* CALL MSTRY (A(1586), A(1511), A(1546), DTIME)
00126 31* RETURN
00127 32* END

END OF COMPIKATION: NO DIAGNOSTICS.
**SAMPLE PROBLEM FOR SINDA VERSION 9**

*FOR TRANCAP*

**UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A - LEVEX LEVEL E12900010008**

This compilation was done on 25 FEB 73 at 19:16:12

**SUBROUTINE OUTCAL**

**ENTRY POINT 000035**

**STORAGE USED:** CODE(11) 000037; DATMC(1) 000006; BLKND COMMON(23) 000000

**COMMON BLOCKS:**

- 0003 TITLE 000024
- 0004 TEMP 000001
- 0005 CAP 000001
- 0006 SOURCE 000001
- 0007 COND 000001
- 0010 PCI 000001
- 0011 PE 000001
- 0012 RST 000001
- 0013 ARRAY 000001
- 0014 FIXCON 000002
- 0015 DIPENS 000011
- 0016 SPACE 000003

**EXTERNAL REFERENCES (BLOCK, NAME):**

- 0017 TPRINT
- 0020 FLPRINT
- 0021 TIMCHK
- 0022 RERR3

**STORAGE ASSIGNMENT (BLOCK, TYPE, RELATIVE LOCATION, NAME):**

| 0013 A 000000 A | 0014 000022 ARILCA | 0014 000035 ARILCC | 0014 000012 ATIPRCA | 0014 000017 ATIPPR |
| 0014 000013 BACKUP | 0014 000040 BARTACI | 0005 000021 C | 0014 000009 CSGRPL | 0014 000026 CSGREQ |
| 0014 000020 CSGCMN | 0014 000027 CSGRPL | 0014 000037 CSGRPL | 0014 000010 DPRA | 0014 000111 DPPC |
| 0014 000031 GRLICA | 0014 000032 GRLICL | 0014 000027 DIPEN | 0014 000024 DIFPEI |
| 0014 000050 DIPEN | 0014 000005 DIPMLP | 0014 000036 DIPMLP | 0014 000027 EILT | 0014 000024 EILT |
| 0015 000000 M | 0000 000001 INPS | 0015 000006 LHFFA | 0014 000037 LSQ | 0014 000055 IEST | 0014 000052 IEST |
| 0014 000036 LSQ | 0015 000006 LHFFA | 0015 000006 LSQ | 0014 000037 LSQ | 0014 000055 IEST | 0014 000052 IEST |
| 0014 000044 MALLTC | 0015 000006 KAT | 0014 000045 MALLTC | 0014 000042 MALLTC | 0014 000044 MALLTC |
| 0016 000000 ADAM | 0014 000043 KATPCC | 0015 000021 MCT | 0015 000043 MCTPCC | 0015 000021 KAM |
| 0015 000000 MARS | 0015 000006 MARS | 0014 000041 MADP | 0016 000011 MASI | 0016 000007 MA |
| 0014 000000 OPETRA | 0014 000001 OPETRA | 0014 000001 OPETRA | 0014 000011 OPETRA | 0016 000011 OPETRA |
| 0014 000000 SESE | 0014 000000 SESE | 0014 000000 SESE | 0014 000000 SESE | 0014 000000 SESE |
| 0014 000015 TIPEN | 0014 000000 TIPEN | 0014 000015 TIPEN | 0014 000015 TIPEN | 0014 000015 TIPEN |
| 0014 000056 TIPEND | 0014 000057 TIPEND | 0016 000052 X | 0014 000057 TIPEND | 0014 000057 TIPEND |

**001**

- SUBROUTINE OUTCAL

**00103**

- COMMON/STATED/ W

**00104**

- COMMON/STEPPS/ T
SAMPLE PROBLEM FOR SINDA VERSION 9

DATE 250273 PAGE 37

END OF COMPILATION: NO DIAGNOSTICS.
STARTING ADDRESS 014000
CORE LIMITS 014000 014426 100000 123456 163772 163777

SINDA CODE
0 100000-100000
1 014000-014513

NSTOPS/ALCS
1 014014-014625

NTCS/ALCS
0 100001-100001
1 014026-014333
2 100002-100004

AFATS/ALCS
1 014334-015271
2 100077-100113

AFATC/ALCS
1 015272-015314

RCHVTS/ALCS
1 015315-015441
2 100814-100822

RCHTS/ALCS
1 015542-016211
2 100223-100246

SPACES/ALCS
1 016212-016255

DEPINC********
0 100247-100256

REARS/ALCS
0 100257-100264
1 016258-016272

REIRA/ALCS
1 016721-016772
2 105045-105475

EXEC/ALCS
0 105476-105512
1 016773-017015
<table>
<thead>
<tr>
<th>CS</th>
<th>CODE</th>
<th>O</th>
<th>106633-106637</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>103391-103398</td>
</tr>
<tr>
<td>EIP</td>
<td>/PL24</td>
<td></td>
<td>035375-035375</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>104660-104670</td>
</tr>
<tr>
<td>REVPOL/</td>
<td>CODE</td>
<td></td>
<td>104733-104733</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>035376-035372</td>
</tr>
<tr>
<td>NWAK</td>
<td>/CODE</td>
<td></td>
<td>104733-105120</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>035673-035677</td>
</tr>
<tr>
<td>FLOKAL/</td>
<td>CODE</td>
<td></td>
<td>1055121-105545</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>036360-040370</td>
</tr>
<tr>
<td>SYMBOL/</td>
<td>CODE</td>
<td></td>
<td>1053926-105623</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>040091-040425</td>
</tr>
<tr>
<td>NFSD</td>
<td>/CODE</td>
<td></td>
<td>1056224-105670</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>040424-040617</td>
</tr>
<tr>
<td>DSORT /PL24</td>
<td></td>
<td>1</td>
<td>0406720-040647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1056721-105707</td>
</tr>
<tr>
<td>CKPP55/</td>
<td>CODE</td>
<td></td>
<td>105710-105752</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>0406790-041149</td>
</tr>
<tr>
<td>GEN3UT/</td>
<td>CODE</td>
<td></td>
<td>105753-105027</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>041141-041535</td>
</tr>
<tr>
<td>PARM</td>
<td>/CODE</td>
<td></td>
<td>106033-106072</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>041536-041667</td>
</tr>
<tr>
<td>FLAMES/</td>
<td>CODE</td>
<td></td>
<td>106079-106423</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>0416750-042321</td>
</tr>
<tr>
<td>SORT  /PL24</td>
<td></td>
<td>0</td>
<td>1042320-104357</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>042320-042361</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>104260-104265</td>
</tr>
<tr>
<td>NWAK1/</td>
<td>CODE</td>
<td></td>
<td>104266-104451</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>042362-043054</td>
</tr>
<tr>
<td>NWAKN/</td>
<td>CODE</td>
<td></td>
<td>104452-104632</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>043085-043553</td>
</tr>
<tr>
<td>PRINTIN/YYYYY</td>
<td></td>
<td>0</td>
<td>104633-104637</td>
</tr>
</tbody>
</table>
DATA /*****
  0 106690-106697

INPUT/COEF
  0 106700-106731
  1 106735-106742

SPACE/*****
  0 106743-112653

LOGIC/*****
  0 112459-112467

DIMENS/*****
  0 112468-112470

FINCON/*****
  0 112471-112472

ARRAY/*****
  0 112473-112671

REPORT/*****
  0 117672-120074

PC2/*****
  0 120075-120546

PC1/*****
  0 120547-121423

COND/*****
  0 121424-122113

SOURCE/*****
  0 122114-122665

CAP/*****
  0 122666-123037

TEPP/*****
  0 123038-123415

TITLE/*****
  0 123416-123441

END OF ALLOCATION 1103 0539A 09099
### Sample Problem for SINDA Version 9

#### Table

<table>
<thead>
<tr>
<th>Time</th>
<th>Time</th>
<th>Cog</th>
<th>Temp</th>
<th>Rel Temp</th>
<th>Flow Rate</th>
<th>Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>Flow Rate</td>
<td>Pressure</td>
</tr>
</tbody>
</table>

#### Flow Rates (L/min)

<table>
<thead>
<tr>
<th>Flow Rate</th>
<th>Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>240.8</td>
<td>244.8</td>
</tr>
<tr>
<td>244.8</td>
<td>248.5</td>
</tr>
<tr>
<td>248.5</td>
<td>252.000</td>
</tr>
<tr>
<td>252.000</td>
<td>255.500</td>
</tr>
<tr>
<td>255.500</td>
<td>259.000</td>
</tr>
<tr>
<td>259.000</td>
<td>262.500</td>
</tr>
<tr>
<td>262.500</td>
<td>266.000</td>
</tr>
<tr>
<td>266.000</td>
<td>269.500</td>
</tr>
<tr>
<td>269.500</td>
<td>273.000</td>
</tr>
<tr>
<td>273.000</td>
<td>276.500</td>
</tr>
<tr>
<td>276.500</td>
<td>280.000</td>
</tr>
<tr>
<td>280.000</td>
<td>283.500</td>
</tr>
<tr>
<td>283.500</td>
<td>287.000</td>
</tr>
<tr>
<td>287.000</td>
<td>290.500</td>
</tr>
<tr>
<td>290.500</td>
<td>294.000</td>
</tr>
<tr>
<td>294.000</td>
<td>297.500</td>
</tr>
<tr>
<td>297.500</td>
<td>301.000</td>
</tr>
<tr>
<td>301.000</td>
<td>304.500</td>
</tr>
<tr>
<td>304.500</td>
<td>308.000</td>
</tr>
<tr>
<td>308.000</td>
<td>311.500</td>
</tr>
<tr>
<td>311.500</td>
<td>315.000</td>
</tr>
<tr>
<td>315.000</td>
<td>318.500</td>
</tr>
</tbody>
</table>

#### Pressure (kPa)

<table>
<thead>
<tr>
<th>Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>214.96</td>
</tr>
<tr>
<td>8016.56</td>
</tr>
<tr>
<td>8295.56</td>
</tr>
<tr>
<td>3529.4</td>
</tr>
<tr>
<td>3529.4</td>
</tr>
<tr>
<td>82.399</td>
</tr>
</tbody>
</table>

---

**Note:** The table above represents a sample problem for SINDA Version 9, detailing the flow rates and pressures involved in a thermodynamic analysis.
### Systems Improved Numerical Differentiating Analyzer

<table>
<thead>
<tr>
<th>SAMPLE PROBLEM FOR SINDA VERSION 9</th>
<th>SINDA</th>
<th>UNIVAC-1108 FORTRAN-V VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>6787F.  (0.59590-0.01)  1.9437  1.9437</td>
<td>1.9437</td>
<td>1.9437  (0.59590-0.01)  1.9437  99.220  82.595</td>
</tr>
<tr>
<td>2029.(9)  2029.(9)  3029.(9)  3029.(9)</td>
<td>82.595</td>
<td>793.83  (0.44830-0.01)  560000-01  1.4936  1.9436</td>
</tr>
<tr>
<td>1.9437  1.9437</td>
<td>2014.(9)</td>
<td>0.129999-06  201.(9)  0.913.2  3.1847</td>
</tr>
</tbody>
</table>

**PRESSURES (LBS/FT\(^2\))**

- 0.13034-06  0.13014-06  69977.  9.420.5
- 491.3  9358.1  6228.7  6229.5  2220.0  2226.4

**VALVE POSITIONS**

- 2226.4  2216.8  2216.8  201.\(9\)  0.0000

**COMPUTER TIME**

- 0.99000  0.99000

**DIVIDE CHECK AT 021506**

<table>
<thead>
<tr>
<th>TIME</th>
<th>1.00000+00 TIMEPU: 5.00000-03 CSMGMIN</th>
<th>771= 2.29961-06 TEPQEC</th>
<th>2401= 2.36745-02 RELQEC</th>
<th>1967= 5.20754-03</th>
</tr>
</thead>
<tbody>
<tr>
<td>T 1</td>
<td>71.036  T 2= 52.868  T 3= 37.721  T 4= 24.790  T 5= 19.120  T 6= 14.211</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 13</td>
<td>71.034  T 14= 52.988  T 15= 37.916  T 16= 25.297  T 17= 14.528  T 18= 14.518</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 19</td>
<td>71.035  T 20= 52.865  T 21= 37.716  T 22= 24.991  T 23= 14.254  T 24= 14.246</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 43</td>
<td>71.033  T 44= 50.706  T 45= 70.321  T 46= 69.966  T 47= 69.611  T 48= 69.618</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 49</td>
<td>71.033  T 50= 70.677  T 51= 70.322  T 52= 69.664  T 53= 69.617  T 54= 69.622</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 55</td>
<td>71.034  T 56= 70.677  T 57= 70.323  T 58= 69.670  T 59= 69.616  T 60= 69.619</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 61</td>
<td>71.034  T 62= 70.677  T 63= 70.323  T 64= 69.670  T 65= 69.618  T 66= 69.619</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 67</td>
<td>71.033  T 68= 70.677  T 69= 70.321  T 70= 69.947  T 71= 69.944  T 72= 69.944</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 73</td>
<td>69.621  T 74= 69.249  T 75= 68.919  T 76= 68.969  T 77= 68.910  T 78= 69.212</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 79</td>
<td>69.621  T 80= 69.249  T 81= 68.920  T 82= 68.919  T 83= 68.920  T 84= 68.920</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 85</td>
<td>69.620  T 86= 69.249  T 87= 68.920  T 88= 68.991  T 89= 68.991  T 90= 68.991</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 91</td>
<td>69.621  T 92= 69.249  T 93= 68.918  T 94= 68.991  T 95= 68.991  T 96= 68.991</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 97</td>
<td>71.032  T 98= 71.529  T 99= 71.021  T 100= 71.222  T 101= 71.431  T 102= 71.431</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 103</td>
<td>71.032  T 104= 71.420  T 105= 71.420  T 106= 71.420  T 107= 71.420  T 108= 71.420</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 109</td>
<td>71.032  T 110= 71.420  T 111= 71.420  T 112= 71.420  T 113= 71.420  T 114= 71.420</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 115</td>
<td>71.032  T 116= 71.420  T 117= 71.420  T 118= 71.420  T 119= 71.420  T 120= 71.420</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 201</td>
<td>71.037  T 202= -1.660  T 203= -1.660  T 204= -1.660  T 205= -1.660  T 206= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 207</td>
<td>71.036  T 208= -1.660  T 209= -1.660  T 210= -1.660  T 211= -1.660  T 212= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 213</td>
<td>71.035  T 214= -1.660  T 215= -1.660  T 216= -1.660  T 217= -1.660  T 218= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 219</td>
<td>71.036  T 220= -1.660  T 221= -1.660  T 222= -1.660  T 223= -1.660  T 224= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 225</td>
<td>71.035  T 226= -1.660  T 227= -1.660  T 228= -1.660  T 229= -1.660  T 230= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 231</td>
<td>71.036  T 232= -1.660  T 233= -1.660  T 234= -1.660  T 235= -1.660  T 236= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 237</td>
<td>71.035  T 238= -1.660  T 239= -1.660  T 240= -1.660  T 241= -1.660  T 242= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T 243</td>
<td>71.036  T 244= -1.660  T 245= -1.660  T 246= -1.660  T 247= -1.660  T 248= -1.660</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

140
<table>
<thead>
<tr>
<th>PRESSURE CROPS</th>
<th>FLOE RATES (LB/HR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T 298= 69.621</td>
<td>T 292= 64.122</td>
</tr>
<tr>
<td>T 297= 71.029</td>
<td>T 298= 71.029</td>
</tr>
<tr>
<td>T 309= 69.618</td>
<td>T 310= 69.618</td>
</tr>
<tr>
<td>T 315= 40.059</td>
<td>T 316= 71.092</td>
</tr>
</tbody>
</table>

FLOW RATES (LB/HR)

<table>
<thead>
<tr>
<th>PRESSURE DROPS (LB/FT**2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T 294.28</td>
</tr>
<tr>
<td>T 379.56</td>
</tr>
<tr>
<td>T 379.56</td>
</tr>
</tbody>
</table>

PRESSURES (LB/FT**2)

<table>
<thead>
<tr>
<th>VALVE POSITIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>T 1.500000 +00</td>
</tr>
</tbody>
</table>

**TIME** = 1.952 MINUTES

**TIME** = 2.000000 MINUTE | **TIME** = 5.000000 -03 | **TIME** = 5.000000 -03 | **TIME** = 1.100250 -03 | **TIME** = 5.100000000 | **TIME** = 1.230460 -04

| PAGE | 3 | 142 |
## FLOW RATES (LB/WA)

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>208</th>
<th>209</th>
<th>210</th>
<th>211</th>
<th>212</th>
</tr>
</thead>
<tbody>
<tr>
<td>204.7</td>
<td>2462.25</td>
<td>074.48</td>
<td>037.84</td>
<td>219.03</td>
<td>217.56</td>
<td>219.95</td>
</tr>
<tr>
<td>1587.5</td>
<td>793.24</td>
<td>037.52</td>
<td>036.22</td>
<td>037.52</td>
<td>036.22</td>
<td>037.52</td>
</tr>
<tr>
<td>220.02</td>
<td>329.72</td>
<td>2462.25</td>
<td>074.48</td>
<td>037.84</td>
<td>219.03</td>
<td>217.56</td>
</tr>
<tr>
<td>397.52</td>
<td>386.21</td>
<td>074.48</td>
<td>037.84</td>
<td>219.03</td>
<td>217.56</td>
<td>219.95</td>
</tr>
</tbody>
</table>

## PRESSURE PADS (LB/FT²)

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>208</th>
<th>209</th>
<th>210</th>
<th>211</th>
<th>212</th>
</tr>
</thead>
<tbody>
<tr>
<td>209.2</td>
<td>-684.3</td>
<td>-684.3</td>
<td>-684.3</td>
<td>-684.3</td>
<td>-684.3</td>
<td>-684.3</td>
</tr>
<tr>
<td>9619.9</td>
<td>599.2</td>
<td>599.2</td>
<td>599.2</td>
<td>599.2</td>
<td>599.2</td>
<td>599.2</td>
</tr>
<tr>
<td>370.61</td>
<td>370.61</td>
<td>370.61</td>
<td>370.61</td>
<td>370.61</td>
<td>370.61</td>
<td>370.61</td>
</tr>
<tr>
<td>3621.8</td>
<td>3621.8</td>
<td>3621.8</td>
<td>3621.8</td>
<td>3621.8</td>
<td>3621.8</td>
<td>3621.8</td>
</tr>
<tr>
<td>6509.4</td>
<td>6509.4</td>
<td>6509.4</td>
<td>6509.4</td>
<td>6509.4</td>
<td>6509.4</td>
<td>6509.4</td>
</tr>
</tbody>
</table>

## PRESSURES (LB/FT²)

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>208</th>
<th>209</th>
<th>210</th>
<th>211</th>
<th>212</th>
</tr>
</thead>
<tbody>
<tr>
<td>12922.7</td>
<td>12922.7</td>
<td>12922.7</td>
<td>12922.7</td>
<td>12922.7</td>
<td>12922.7</td>
<td>12922.7</td>
</tr>
<tr>
<td>16060.0</td>
<td>16060.0</td>
<td>16060.0</td>
<td>16060.0</td>
<td>16060.0</td>
<td>16060.0</td>
<td>16060.0</td>
</tr>
</tbody>
</table>

## VALVE POSITIONS

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>208</th>
<th>209</th>
<th>210</th>
<th>211</th>
<th>212</th>
</tr>
</thead>
<tbody>
<tr>
<td>2422.2</td>
<td>2422.2</td>
<td>2422.2</td>
<td>2422.2</td>
<td>2422.2</td>
<td>2422.2</td>
<td>2422.2</td>
</tr>
</tbody>
</table>

## COMPUTER TIME = 2.922 MINUTES

---

### Notes

- Flow rates and pressure pads are measured in pounds per hour per square inch (LB/WA).
- Pressure readings are given in pounds per square foot (LB/FT²).
<table>
<thead>
<tr>
<th>FLOW RATES (lb/hr)</th>
<th>2465.7</th>
<th>2962.2</th>
<th>874.70</th>
<th>874.70</th>
<th>37.40</th>
<th>219.89</th>
<th>217.52</th>
<th>219.82</th>
<th>217.46</th>
<th>37.29</th>
<th>674.68</th>
<th>347.83</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRESSURE DROPS (lb/ft^2)</td>
<td>554.19</td>
<td>260.22</td>
<td>347.52</td>
<td>396.21</td>
<td>397.52</td>
<td>396.21</td>
<td>397.52</td>
<td>396.21</td>
<td>397.52</td>
<td>396.21</td>
<td>397.52</td>
<td>396.21</td>
</tr>
<tr>
<td>PRESSURES (lb/ft^2)</td>
<td>204.28</td>
<td>6294.9</td>
<td>65468</td>
<td>13.887</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
<td>379.23</td>
</tr>
<tr>
<td>VALVE POSITIONS</td>
<td>.99950</td>
<td>.34220</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SAMPLE PROBLEM FOR SINDA VERSION 9

COMPUTER TIME = 3.886 MINUTES

END OF DATA
FIGURE 12
RADIATOR TEMPERATURES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM
[1] RADIATOR INLET TEMPERATURE -- DEGF
[2] MAIN RADIATOR OUTLET TEMPERATURE -- DEGF
[3] PRIME TUBE OUTLET TEMPERATURE -- DEGF
[4] MIXED RADIATOR OUTLET TEMPERATURE -- DEGF
FIGURE 13
SYSTEM TEMPERATURES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM

[1] RADIATOR CONTROLLED OUTLET, HX INLET -- DEGF
[2] HX OUTLET ON RADIATOR SIDE -- DEGF
[3] HX INLET ON WATER SIDE -- DEGF
[4] HX OUTLET ON WATER SIDE -- DEGF

TIME - (HOURS)

TEMPERATURE

RADIATOR CONTROLLED OUTLET, HX INLET
FIGURE 14

SYSTEM PRESSURES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM

[1] TOTAL PUMP FLOW RATE -- LB/HR
[2] TOTAL RADIATOR FLOW RATE -- LB/HR
[3] BYPASS FLOW RATE -- LB/HR

TIME - (HOURS)
FIGURE 15
RADIATOR FLOW RATES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM

1. TOTAL RADIATOR FLOW RATE -- LB/HR
2. MAIN RADIATOR FLOW RATE -- LB/HR
3. PRIME TUBE FLOW RATE -- LB/HR

FLOW RATE

TIME - (HOURS)

0 1 2 3

0 500 1000 1500 2000 2500
FIGURE 16
PUMP PRESSURES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM

[1] PUMP OUTLET PRESSURE -- PSF
[2] VALVE 1 INLET PRESSURE -- PSF
[3] VALVE 2 INLET PRESSURE -- PSF
[4] PUMP INLET PRESSURE -- PSF

TIME - (HOURS)
FIGURE 17

SYSTEM PRESSURES FOR SAMPLE PROBLEM

SINDA VERSION 9 SAMPLE PROBLEM

[1] MAIN RADIATOR INLET PRESSURE -- PSF
[2] PRIME TUBE INLET PRESSURE -- PSF
[3] PRESSURE AT RADIATOR OUTLET -- PSF
[4] PRESSURE AT HX INLET -- PSF

PRESSURE

10000
8000
6000
4000
2000
0

TIME - (HOURS)

0 1 2 3
6.0 REFERENCES


APPENDIX A

SUBROUTINE LISTINGS

A Fortran listing is presented below for the subroutines which were modified or added to the SINDA preprocessor and user subroutine library which create SINDA/VERSION 9.
CALL SREADC(1)
IF (MCMNE.COMMT) GO TO 10
WRITE (OUT, 100) FOR BLANK, ALPH, COL1
GO TO 10
10 CONTINUE
IF (ALPH(3).EQ.ENDOF) GO TO 520
IF (ALPH(3).EQ.ENDOF) GO TO 43
WRITE (OUT, 201)
WRITE (OUT, 403) ALPH
DEBUG PRINT IF * IN COLUMN 80
IF (ALPH(41).EQ.PRINT) PRINT* TRUE.
IF (ALPH(3).EQ.THERMO) GO TO 60
THERMAL PROBLEM - CHECK FOR LONG OR SHORT PSEUDO COMPUTE SET.
IF (ALPH(41).EQ.THERMO) GO TO 30
LONG* TRUE.
IF (ALPH(1).EQ.(W) LONG*, TRUE.
GO TO 80
30 IF (ALPH(41).EQ.PCSHMT) GO TO 350
GO TO 80
CHECK FOR INITIAL PARAMETER RUN
40 CONTINUE
IF (ALPH(3).EQ.PCSHMT) GO TO 30
IF (ALPH(3).EQ.PCSHMT) GO TO 30
IF (ALPH(3).EQ.PCSHMT) GO TO 30
FINAL PARAMETER RUN
50 IF (ALPH(3).EQ.FINES) GO TO 310
IF (ALPH(3).EQ.FINES) GO TO 310
IF (ALPH(3).EQ.FINES) GO TO 310
CHECK FOR GENERAL PROBLEM
60 CONTINUE
60 CONTINUE
IF (ALPH(3).EQ.GENERAL) GO TO 40
IF (ALPH(3).EQ.GENERAL) GO TO 40
LOC(1)=0
70 CONTINUE
10 CONTINUE
SET UP TITLE
80 CONTINUE
N=0
J=1
90 CALL SREADC(2)
90 CALL SREADC(2)
IF (MCNNE.COMMT) GO TO 100
WRITE (OUT, 403) BLANK, ALPH, COL1
GO TO 40
312. WRITE (OUT,673) BLANK,Col27,ALPH,Col1
313. IF (COL(1).EQ.COMMIT) GO TO 290
314. IF (ALPH(1).EQ.REMARK) GO TO 280
315. IF (ALPH(1).NE.ARRAY) GO TO 210
316. IF (ALPH(1).EQ.PRINT) ARRAY=.TRUE.
317. 290 CONTINUE
318. CALL SPREAD (3)
319. WRITE (OUT,670) BLANK,Col27,ALPH,Col1
320. IF (COL(1).EQ.COMMIT) GO TO 290
321. IF (ALPH(1).EQ.END) GO TO 310
322. RETURN
323. LNODE=.FALSE.
324. LEND=.FALSE.
325. LEONE=.FALSE.
326. LARRAY=.TRUE.
327. LOC(13)=LOC(12)+LEN(12)
328. LEND=LOC(13)-LEN(13)
329. LOC(14)=LOC(13)+200
330. LEN(14)=0
331. LOC(15)=LOC(14)+200
332. LEN(15)=0
333. CALL DATA0
334. CALL DATA1
335. 300 CONTINUE
336. CALL SQUEEZE (13,15)
337. CALL WRITE (4)
338. CALL WRITE (4)
339. IF (I.NET.PLINT) GO TO 310
340. WRITE (OUT,740) LEN
341. WRITE (OUT,700) (:,LOC(11),LEN(11),LEN(13),135)
342. M=LOC(11)
343. M=LOC(11)+LEN(11)-1
344. WRITE (OUT,710) (:,LEN(11),LEN(11),LEN(M),
345. C)
346. C NORMAL RETURN
347. C
348. 310 CONTINUE
349. IF (J.NET.GO TO 320
350. SET LOC AND LLEN FOR CALL TO PSEUD0
351. M=LEN(11)-LOC(15)+LEN(15)+1
352. M=LEN(M)/2
353. LOC(I&)+LOC(15)-LEN(15)
354. LEN(I&)=0
355. LOC(I&)+LOC(16)+INC
356. LEN(I&)=0
357. CONVET ARAYS AND CONSTATNS TO FORTRAN
358. 320 CONTINUE
359. M=I+
360. J=I
361. 330 IF (LEN(I).EQ.0) GO TO 400
362. K=LOC(I)
363. 340 KEND=K+LEN(I)
364. 340 TYPE=LOC(2,0,0,0)
365. FLOD=5,FLOD=0,FRED=0,FRED=5
366. F_TYPE=LOC(2,0,0,0)
367. 340 IEND=K
368. IF (I.NET.GO TO 320
369. M=LOC(I+1)END)
370. M=LOC(I+1)END)
371. RETURN
372. CALL RELATE (2,1,0,0,0}
373. GO TO 360
374. GO TO 360
375. GO TO 360
I1. BLOCK DESIGNATION ENCOUNTERED .

440 FORMAT (AM = A.9DH THE PSEUDOCOMPLETE SEQUENCE INDICATOR MUST BE DELETED)

441. I EITHER SPC5 OR LPCS, AND START IN COLUMN 21

442. PDD FORMAT (H CHAR USE LOC AND LEN,1000100)

443. TID FORMAT (12H CHAR SEQUENCE,110,10,205)

444. 725 FORMAT (AH AND,16,AH AND,16,AH AND,16,AH AND,16)

445. 726 FORMAT (AVG,16,AVG,16,AVG,16,AVG,16)

446. TID FORMAT (AVG,16,AVG,16,AVG,16,AVG,16)

447. 726 FORMAT (AVG,16,AVG,16,AVG,16,AVG,16)

448. D10 FORMAT (12H FIXED CONSTANTS ARRAY,110,125,255,85,3123)

449. D11 FORMAT (VH LEN,181

451. END
SUBROUTINE IMBED

COMMON /BUCKET/ IB(1)
COMMON /DATA/ DMA(4), NUC, NUM, NUM(4), ENDATA

DIMENSION KEY(1/2, 2H, 2H, 2H, 2H)

LOGICAL CRDERR

KEY = 6H

ARRAYS

I 100

100 LL = 1

IST = LOC(14)

IEND = IST + LEM(14) - 1

DO 140 J = IST, IEND

IF (NUM(J, 2) .EQ. IB(13)) GO TO 390

140 CONTINUE

DO TO 380

GO TO 360

GO TO (100, 200, 300, 250)

CONSTANTS

100 NUM = IB(11)

CALL CONVRT(12, 30, NUM, CRDERR)

IF (CRDERR) 60 TO 380

CRDERR = 0

30 TO 190

C TEMPERATURES

100 NLOC = LOC(13)

NCEN = LEN(13)

60 TO 360

CONDUCTORS

100 NLOC = LOC(15)

NLEN = NGT

IF BED 89.

C
360 CALL SEARCHNUM,(BINLOC),NLEN,L1
361 IF(L1) 360,N99,390
380 ERDATA = 1.0
381 IF(L) 3830,3,90
382 WRITE(6,383) EN11,NN,IBM
383 IF(IBM) 386,385
384 FORMAT(99,24H referenced at location 15, 25H is not in the list ** *)
385 IMBED
386 l9H OF ARRAY 15, 26H IS NOT IN THE LIST ** *)
387 GO TO 00
390 IF(1) = L
400 CONTINUE
409 CONTINUE
410 END
1. FUNCTION ABS1(X)
2.   ABS1 = ABS(X)
3. RETURN
4. END
SUBROUTINE CABIN(NLOC,IC,SUMOL,SMOL)

LOGICAL EXPCT

COMMON /ARRAY/ DATA(I)
COMMON /SEC/ TIME(1)
COMMON /FREQ/ HFP(1)
COMMON /SPACE/ ATM, NIN, ERT(I)
COMMON /FREQ/ AND, NAA

DIMENSION NLOC(I)
DIMENSION NDATA(I)
DIMENSION NECT(I)

EQUIVALENCE (CONI1),TIME(I), (CONI2),TINC(I), (CONI22),BTIME(I)
EQUIVALENCE (DATA, NDATA), IEIT, NEIT)

DEFINE (DATA(I) = EXT(NN+1)

***

DEFINE ETAUC(I) = ETAUC(I)

IF(DTIME(I).GT.0.0) EXPCT = .FALSE.

IF(NLOC(I).EQ.6) GO TO 102

CALL TOPLIN

WRITE(6,101) NLOC(I)

101 FORMAT(5HO.* INCORRECT NUMBER OF ELEMENTS INPUT TO CABIN, IC

I=15,

CALL WLBCK

CALL EXIT

102 NST = NLOC(Z)

NCRV = NLOC(3)

NCON = ALOC(q)

LHC = NLOC(S)

LHFP = NLOC(6)

LHTB = NLOC(T)

NCH = NTH - I

NSPT = 0

NL2 = 0

NL3 = 0

IF(LHTB.GT.0) NL1 = NDATA(LHTB)

IF(LHFP.GT.0) ML2 = NDATA(LHFP)

IF(LHC.GT.0) ML3 = NDATA(LHC)

NSPT = (NL1/4 + ML2/5 + NL3/2) * 3

MODLANS = NSPT

IF(NL2.GT.0) NSPT = 0

NEED = NSPT - NIM

IF(NL2.GT.0) NSPT = 0

NEED = NSPT - NIM

CALL TOPLIN

WRITE(6,103) NEED

103 FORMAT(5H0. * INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR CABIN

ANALYSIS SUBROUTINE * * * // EX SHORT 15, 1'M LOCATIONS)

CALL WLBCK

CALL EXIT

104 CONTINUE
61. AS = NDATAINST+1
62. NVT = NT(NDATAINST)) - 1
63. IF (NDATAINST .LT. 0) GO TO 2
64. IF(V .GE. MV1/2) CALL ERR(NST)
65. NDATAINST = NDATAINST
66. IF (NDATAINST .LT. 0) GO TO 2
67. IF(V .GE. MV1/2) CALL ERR(NST)
68. NDATAINST = NDATAINST
69. IF(V .GE. MV1/2) CALL ERR(NST)
70. ICABIN = ICABIN
71. ICABIN = ICABIN
72. ICABIN = ICABIN
73. ICABIN = ICABIN
74. ICABIN = ICABIN
75. ICABIN = ICABIN
76. ICABIN = ICABIN
77. ICABIN = ICABIN
78. ICABIN = ICABIN
79. ICABIN = ICABIN
80. ICABIN = ICABIN
81. ICABIN = ICABIN
82. ICABIN = ICABIN
83. ICABIN = ICABIN
84. ICABIN = ICABIN
85. ICABIN = ICABIN
86. ICABIN = ICABIN
87. ICABIN = ICABIN
88. ICABIN = ICABIN
89. ICABIN = ICABIN
90. ICABIN = ICABIN
91. ICABIN = ICABIN
92. ICABIN = ICABIN
93. ICABIN = ICABIN
94. ICABIN = ICABIN
95. ICABIN = ICABIN
96. ICABIN = ICABIN
97. ICABIN = ICABIN
98. ICABIN = ICABIN
99. ICABIN = ICABIN
100. ICABIN = ICABIN
101. ICABIN = ICABIN
102. ICABIN = ICABIN
103. ICABIN = ICABIN
104. ICABIN = ICABIN
105. ICABIN = ICABIN
106. ICABIN = ICABIN
107. ICABIN = ICABIN
108. ICABIN = ICABIN
109. ICABIN = ICABIN
110. ICABIN = ICABIN
111. ICABIN = ICABIN
112. ICABIN = ICABIN
113. ICABIN = ICABIN
114. ICABIN = ICABIN
115. ICABIN = ICABIN
116. ICABIN = ICABIN
117. ICABIN = ICABIN
118. ICABIN = ICABIN
119. ICABIN = ICABIN
120. ICABIN = ICABIN
121. ICABIN = ICABIN
122. ICABIN = ICABIN
UV = POL(NDATA(NCRV*33),TC)
CPA = POL(CPA,TC)
CPV = POL(CPV,TC)
CN = POL(NDATA(NCRV*33),TC)
CV = POL(NDATA(NCRV*33),TC)

UC = EXC+CV*PSIEAB+Cu+EXC+PSIEAB
CABIN 128.

CABIN 129.
CPV = POL(NCPV,TC)

CABIN 130.
CC = CPA*PSICAB*CPV/(1.0*PSICAB)

CABIN 132.

CABIN 133.

CABIN 134.

CABIN 135.

CABIN 136.

CABIN 137.

CABIN 138.

CABIN 139.

CABIN 140.

CABIN 141.

CABIN 142.

CABIN 143.

CABIN 144.

CABIN 145.

CABIN 146.

CABIN 147.

CABIN 148.

CABIN 149.

CABIN 150.

CABIN 151.

CABIN 152.

CABIN 153.

CABIN 154.

CABIN 155.

CABIN 156.

CABIN 157.

CABIN 158.

CABIN 159.

CABIN 160.

CABIN 161.

CABIN 162.

CABIN 163.

CABIN 164.

CABIN 165.

CABIN 166.

CABIN 167.

CABIN 168.

CABIN 169.

CABIN 170.

CABIN 171.

CABIN 172.

CABIN 173.

CABIN 174.

CABIN 175.

CABIN 176.

CABIN 177.

CABIN 178.

CABIN 179.

CABIN 180.

CABIN 181.

CABIN 182.

CABIN 183.

CABIN 184.

CABIN 185.
SUBROUTINE CONCK(N,NL,INDO,IND1)
  DO 60 I=1,NL,INDO
  LL = LL + 1
  PWI = E(TLL)
  IF(PWPME-PWPME-LT. 0.) PW = PWI/PWPME/PWPME
  IF(E(T(KK))LT. 0.) E(T(KK)) = -E(T(KK))
  SUMOL = SUPV*DWI
  OL = OL + OL
  SUMOL = SUMOL + OL
  PWI = PWI + PWI
  E(T(LL)) = PWI
  SUMOL = SUMOL + OL
  RETURN
60 CONTINUE
RETURN
SUBROUTINE OSUM
  IF(EXPLCT) DTAU(J) = DTAU(J) + NA
  OL = OL + OL
  SUMOL = SUMOL + OL
  E(T(LL)) = OL
  SUMOL = SUMOL + OL
  RETURN
END
SUBROUTINE EMAPSS(SPR,A,B,L28,6)

THIS SUBROUTINE REDUCES THE COEFFICIENT MATRIX FOR PFCs

DIMENSION A(L1), B(L1), L28(L1)

LOCATE RELATIVE PRESSURE NODE NUMBER (NPR) OF ACTUAL PRESSURE

NODE NUMBER (IPR) WHICH HAS A SPECIFIED PRESSURE (SPR)

MPRN = L28(1)  
DO 9 NPR = 1,MPRN
   IF(IPR .EQ. L28(NPR+1)) GO TO 10

% CONTINUE
RETURN 6

CALCULATE THE STARTING LOCATION OF COLUMN NPR

10 LOC = (NPR-1)+NPR/2
   LD = LOC + NPR + NPR

MODIFY THE RIGHT HAND SIDE

DO 20 J=I,PRN
   IF(J .EQ. IPR) 12,20,15
      B(J) = B(J) - SPR*ALOC+J)
   GO TO 20

CONTINUE

CALCULATE THE STARTING LOCATION OF ROW NPR

LD = LOC + NPR
   NPR = NPR - 1

DO 30 J=I,PRN
   L28(J) = L28(J*I)

CONTINUE

DELETE COLUMN NPR

IF(IPR .EQ. NPR) GO TO 25
   L28(I) = L28(I)+1

DO 25 L=I,J
   L28(L) = L28(L)+1

CONTINUE

DELETE ROW NPR

IF(I .EQ. IPR) GO TO 35
   L28(I) = L28(I)+1

DO 35 J=I,PRN
   L28(J) = L28(J)+1

CONTINUE

DELETE ACTUAL PRESSURE NODE IPR FROM THE LIST OF ACTUAL PRESSURE NODES

DO 40 J=I,PRN
   L28(J) = L28(J+1)

CONTINUE
SUBROUTINE CNFAST

C AN EXPLICIT EXECUTION SUBROUTINE FOR SINDA FORTRAN V

C THE SHORT PSEUDO COMPUTE SEQUENCE IS REQUIRED

C NODES WITH EIG BELOW OPFPM E RECEIVE STEADY STATE SOLUTION

C NO BACKING UP IS DONE OR ALLOWED

INCLUDE COMP,LIST
INCLUDE DEF,LIST
INCLUDE VARC,LIST
INCLUDE VAR0,IIST

C KON(51),I.E.0) KON(51) = 1
C KON(103),I.E.0) KON(103) = 1.E+0
C KON(103),I.E.0) KON(103) = 1.0
C KON(181),I.E.0) KON(181) = 999
C KON(199),I.E.0) KON(199) = 1.E+0
C KON(211),I.E.0) KON(211) = 0.12
C KON(311),I.E.0) KON(311) = 995

PASS = -1.0
NRC = NMA+MND
IE = NTH
NLA = NDIM
NTH = NTH+MND
NDIM = NDIM+MND
IF(NDIM.LT.0) GO TO 997
NDIM = NNO.ID
TPRINT = CON(13)
TSTEP = CON(21)
TSTEP = 0.0
TSTEP = CON(21)+0.000001
TSTEP = CON(21)-TSUM
TSTEP = TSTEP+TSUM
TPRINT = TSTEP*TSUM+TSTEP
TPRINT = TPRINT+TSTEP
DE(LE) = 0.0
CONTINUE
KON(12) = 0
CALL VARB1
KON(12) = 0
IF(PASS.GT.0.) GO TO 95
PASS = 1.0
CON(1) = TPRINT*TSTEP
CON(1) = TSTEP
IF(TSUM#2.0*TSTEP.GT.CON(18)) TSTEP = 0.5*(CON(18)-TSUM)
CONTINUE
J1 = 0
J2 = 1
JT = 1.4
IE = IE+1
INCLUDE VARP,LIST
INCLUDE VAR2,LIST
T3.
II
T(LA)*460.0
CIAST
612.
V
6(LG).(T1+T-T1ZI*(T1*T2I)
COO
E A
FAST
125.
TO TO
120
ChFrST
1?6. 115 Gv
2
C(LG)
CFAST
128.
20
S
UrPC
Se'C*GV
CNFAST
129.
Su
CV
*5MCGVTZ
ATFASI
130.
C
CHECK
FOR
LAST
CONDUCTOR
TO
THIS
NODE
CNFAST
131.
IF(INStI(JJI ).GT.)
3 TO 110
•
CNFAST
132.
II
;CAFPN(SUPCVeO(L))/SUMC*DAPO*T(ILI
CNFAST
33).
AS(T(L)-Ti)
CNFAST
34.
IFC(RL.GE.2T)
GO
TO
190
CNFAST
135.
RL = T1
CNFAST
136.
KON(3)
L
CNFAST
137.
190
T(L)
Ti
CNFAST
138.
CONTINUE CNFAST
139.
IF(RL1.LE.CON
19))
GO
TO
155
CNFAST
140.
CONTINUE CNFAST
141. CON(30) z RLI CNFAST
192.
160
CALL VARIL2
CNFAST
43.
CON(I3) = CON(1)
CNFAST
1%.
TSUm
I
TSUM*TSUMEASECNFAST
146.
IFITSUM.LT.CON(18))
GO
TO
10
CNFAST
1D9.
TPRINT = TPRINT*TSUM
CNFAST
140.
CALL OUTCAL
CNFAST
199.
IF(CON()*I.00000.LT.CON(3)) GO TO
5
CNFAST
10.
ITH
: IE
CNFAST
11.
NDIM = NLA
CNFAST
152.
RETURN
CNFAST
995 WRITE(6,885)
CNFAST
195.
GO TO 1000
CNFAST
996 WRITE (6,885)
CNFAST
195.
GO TO 1000
CNFAST
997 WRITE(6,885) NDIM
CNFAST
195.
GO TO 1000
CNFAST
998 WRITE (6,885)
CNFAST
160.
GO TO 1000
CNFAST
161.
999 WRITE(6,885)
CNFAST
162.
1000 CALL OUTCAL
CNFAST
163.
CALL EXIT
CNFAST
164. 886 FORMAT(4AM CNFAST REQUIRES SHORT PSEUDO-COMPUTE SEQUENCE)
CNFAST
165. 886 FORMAT(4MH CSH ZERO OR NEGATIVE)
CNFAST
166. 887 FORMAT(8,2H LOCATIONS AVAILABLE)
CNFAST
167. 888 FORMAT(10M N2 OFFPEL)
CNFAST
168. 889 FORMAT(18H NO OUTPUT INTERVAL)
CNFAST
169. END
CNFAST
SUBROUTINE CORBIN(NTAPE, IT, INC, IUNIT)

DIMENSION NBUF(21), ALPHA(I), XSTART(I), XSTOP(I)

IF(NTAPE.GT.0)

READ(6,12) (NTAPE, IUNIT)

NTAPE = -NTAPE

READ(6,13) (CONTINUE, IUNIT, NTAPE)

IF(IT = 0)

IUNIT = IUNIT + 1

DO 20 1 = 1, NTAPE

READ(M,5) (DATA(J), J = 1, NBUF)

NTOTAL = NTOTAL + NBUF

WRITE(6,31) (DATA(J), J = 1, NBUF)

WRITE(6,32) (DATA(J), J = 1, NBUF)

DO 20 1 = 1, NTAPE

READ(M,5) (DATA(J), J = 1, NBUF)

NTOTAL = NTOTAL + NBUF

WRITE(6,31) (DATA(J), J = 1, NBUF)

END

FILE KT

REWIND KT
39 CONTINUE
40 IF (K.LE.15) WRITE(6,92) NTAPE,ALPHA(T)
42 FORMAT(10HDATA FROM 17,32H TAPES HAS BEEN COMBINED ON UNIT A2)
43 RETURN
44 END
CONOTI

1. SUBROUTINE CONOTI(NLOC)

2. C DIMENSION NLOC(I)

3. COMMON ARRAY / NDATA(I) /

4. C COMMON FDATA / F (I) /

5. C COMMON FDATA / G (I) /

6. C COMMON FDATA / CM13 /

7. COMMON FDATA / CM20 /

8. C IFMODMLOC(I,4) .EQ. 03 GO TO 20

9. CALL TOPLIN

10. WRITE(*,101) NLOC(I)

11. 10 FORMAT(F10.0) ** INCORRECT NUMBER OF ELEMENTS INPUT TO CONOTI FOR

12. CONDUCTION DATA, IC = 15, TH = ++ 

13. CALL WRITE

14. CALL EXIT

15. C

16. 20 IC = NLOC(I)

17. DO 100 I = 1, IC,N

18. NG = NLOC(I + 1)

19. HL = NLOC(I + 2)

20. MTE = NLOC(I + 3)

21. MTEP = NLOC(I + 4)

22. IF(MTE .LT. MTEP) GO TO 40

23. IF(NG .LT. MTEP) GO TO 80

24. CALL TOPLIN

25. WRITE(*,60) (NLOC(I*J),J = 1, N#)

26. 60 FORMAT(55H04.9)

27. CALL DICEGI(T(NLT),NDATA(NTEmP),FTErP)

28. CALL DICEGI(T(NLT),NDATA(NTEmP),FTErP)

29. G(NG1 = T1,TE*FTErMP)

30. 100 CONTINUE

31. RETURN

32. END

CONOTI
SUBROUTINE CONDT2(NLOC)

DIMENSION NLOC(1)

COMMON /ARRAY/ NDATA(11,3)
COMMON /DIMENS/ ND, NNT, NNT
COMMON /COND/ B (11,3)

EQUIVALENCE (NTEMP,FTEMP)

I = NLOC(1)

DO 10 1 = 1, IC, 3

NG = NLOC(1+I)

NTIME = NLOC(I+2)

FTEMP = NLOC(I+3)

IF(NG .LE. NGT) GO TO 80

CALL TOPLIN

WRITE(6,10) (NLOC(I*J), J = I, 3)

10 FORMAT(55H00
2IF(NG .LT. NGT) GO TO 10

CALL TOPLIN

WRITE(6,90) (NLOC(I*J), J = I, 3)

90 FORMAT(30H00
3IF(NG .LT. NGT) GO TO 30

CALL WKBCK

CALL EXIT

CALL TOPLIN

CALL DIOEGI(COND(),NDATA(NTIME),FTIME)

SNG = FTIME*FTEMP

CONTINUE

RETURN

END
SUBROUTINE CONV(LLOC,LRO,MLLC)

C
LOGICAL LCP, LMU, LTC

DIMENSION (LLOC(I), MLOC(I), NLOC(I))

COMMON /ARRAY/ NDATA(I)

COMMON ITEM, /T(/)

COMMON /COND/ NND, NNA, NNT, NGT

EQUIVALENCE (ROATA, NDATA)

EQUIVALENCE (NAHT, AHT), (NX, I), (INFI, F2)

C

IF(LLOC(I).EQ. 8) GO TO 20

CALL Toplin

WRITE(6,11) LLOC(I)

11 FORMAT(5HO.A=16, 6H.APR = 16, 6H.AFR = 16)

CALL WRBCK

CALL EXIT

C

00 NT = 0LOC(Z)

IF(NX.LT. I .OR. NX.GT. MAX(I)) GO TO 82

LCP = .TRUE.

C

80 NX = LLOC(I)

IF(NX.LT. 1 .OR. NX.GT. MAX(I)) GO TO 82

LCP = .TRUE.

C
10
61.

6. GO TO 84,
L
02 LCP : .FALSE.

62.
63.

89

64.
65.

MPU

61.
68.

I

,

10.

"AI)

CONVI

I

CONVI

CONVI
CONVI
CONVI
CONVI

IF(NI .LT. I .OR. YN .GT. FAlI)

72.
13.

LTC = .TRUE.
NTC = NI
GO TO 94
S.

C

GO TO 92

CONVI

CONVI
CONVI
CONVI

92 LTC = .FALSE.
TC
=

99 IF(MMO(NLOC(I),)

19.

CALL TOPLIN

80.
61.
62.

WRITE16,95)

CONVI
CONVI

CONVI
CONVI

.EO. 0) GO TO 100

CONVI

NLOCII)
95 FO3MATl(IO. * * INCORRECT NUMBER OF ELEMENTS INPUT TO CONVI FOR
ICONVECTION DATA, (C 15, TH
*)

83.

84.
5.
66.
81.
68.
69.
90.
91.

GO TO 86

.TRUE.

It.

?t.
TO.

CONVI
CONVI
CONVI

.I.

50 T0 88
86 LMU : .FALSE.
)MU
08 41 = ILOc(51

69.

1S.
16.

CONVI

CONVI

CP
aI
l = PILOCIR
IF(NI .LT. I .OR. NI .T.

LU

66.

:

CONVI
CONVI
CONVI

CALL WtIKBCK

CONVI

CALL EXIT

CONVI
CONVI
CONVl
CONVI
CONVI
CONVI
CONVI
CONVI

C
100 IC * NLOC(1)
DO 220 =I:1,C,
AG
: NLOC(I*)
WANT = NLOC(|*2)
ITUBE
P=
NLC( 13)
N*L = NLOC( I4)

92.

ITYPE

93.
91.
95.
96.
97.
98.
99.
100.

= NLOC(I+S

CONVI

Nx

: NLOC(I*6)
NFI
= NLOC(1*7)
NF2
= NLOC(I8)
LTYPE = ITYPE.6 - 5
IF(NG
.GT. NGT) GO TO
IFINFL
.GT. NNT) GO TO
IF(ITUBE .GT. NDATA(L2))
IF(LTYPE .LT. NDATA(LT))

CONVI
CONVI
CONVI
CONVI
CONVI
CONt
CONVI
CONVI

110
110
GO TO 110
GO TO 170

101.

110 CALL TOPLIN

CONVI

102.
103.
109.
105.
106.
101.
108.
109.
110.
Ill.
112.
113.
114.
11t.
116.
117.

ITE(6,120)
120 FORMAT(5HO- • . ERROR IN CONVECTION DATA INPUT TO CONVI
*))
150 VRITE(6,1 0) (NLOC(I+J),J=1,8)
160 FORMAT(SH N = 14, 6H AHT = G63.8, 8H ITUBE : 0, 6H NFL = IS,
I eH ITYPE = 14, qH I = 613.8, SH FI = G13.8, SH F2 GI3.8)
CALL WLKBCK
CALL EXIT

CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI
CONVI

C
170 LTYPE = Li * LTYPE
IP = RDATA(LTYPE)
CSA : RCATA(LTYPE*I)
IF(LCP) CALL D1IDG1IT(NFL),NDATA(NCP), CP)
IF(LUL) CALLDIOEGI( T(NFL ,NCATA(NMU),MU)
IF(LIC) CALL DIDEGl(T(NFL),NDATA(NTC),
TC)
RE = 4.0PABS(ROATA(L2+ITL-BE))YMV/,P
PR = IRMUCP/TC

*NEW
*-I

118.

0

119.

IF(RE .GT. 2000.0) 63 TO 180

CONVI

120.

TEMP = I/ICIE/PR

CONVI

121.

H = TC/D-(3.66.FI *(0.0155.FZ/(TEMP*.0.15*CBPf(TEMP))))
GO 1) 200

CONVI
CONVI

12?.

= %.O0*CSA'P

CONVI


SUBROUTINE CONV2LLOC, MLOC, NLOC)

LOGICAL LCP, LNO, LTP

DIMENSION LLOC(1), MLOC(1), NLOC(1)

DIMENSION NDATA(1)

COMMON ARRAY, NDATA(1)

COMMON STEP / T / (1)

COMMON SCON / C / (1)

COMMON SIMEN / X, NNA, WNT, NGT

EQUIVALENCE (DATA, NDATA)

EQUIVALENCE (NANT, ANT), (NEX, X)

DATA PAXI /65000/

IF(LLOC(1), E8, 69) GO TO 20

CALL TOPLIN

WRITE(10,101) LLOC(1)

20 FORMAT(50) = INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR

IFLOW DATA, IC = 15, 7H ***

CALL WRLCK

CALL EXIT

21. C

22. 20 L2 = LLOC(2)

23. LT = LLOC(7)

24. C

25. IF(LDATA(L2) GT, 0) GO TO 40

26. CALL TOPLIN

27. WRITE(30, NDATA(L2))

28. 30 FORMAT(50) = INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR

IFLOW RATES, IC = 15, 7H ***

GO TO 60

29. C

30. 40 IF(LDATA(L7), E8, 0) GO TO 76

31. CALL TOPLIN

32. WRITE(6503) NDATA(L7)

33. 50 FORMAT(50) = INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR

IFLUID TYPE DATA, IC = 15, 7H ***

60 WRITE(30, PLOC(1,1,2,7))

61. 70 FORMAT(50, 6H FRK = 16, 6H NAF = 16, 6H NAP = 16,

62. 1H NFR = 16, 6H NRT = 16)

63. CALL WRLCK

64. CALL EXIT

65. C

75 IF(LLOC(1), E8, 51) GO TO 60

76. CALL TOPLIN

77. WRITE(3, PLOC(1))

78. 76 FORMAT(50) = INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR

IFLUID PROPERTIES, IC = 15, 7H ***

83. CALL WRLCK

84. CALL EXIT

85. C

80 NX = PLOC(2)

81. IF(NX .LE. 1 .OR. NX .GT. PAXI) GO TO 82

82. LCP = .TRUE.

83. NCP = N3
GO TO 84
82 LCP = .FALSE.
83 CP = X
84 NX = NLOC(I)
85 IF (IN .LT. 1 OR. IN .GT. NMAX) GO TO 86
86 LPU = .FALSE.
87 RPU = NX
88 GO TO 89
89 LPU = .TRUE.
90 RPU = NX
91 GO TO 92
92 LPU = .FALSE.
93 IC = X
94 IF (IN .GT. 1 OR. IN .LT. I) GO TO 93
95 WRITE(6,95) NLOC(I)
96 95 FORMAT(5HO, * * INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 *)
97 IC = 15, IN = 9
98 CALL WPBK
99 CALL EXIT
100 IC = NLOC(I)
101 GO TO 20
102 NC = NLOC(I-1)
103 NMAX = NLOC(I+1)
104 ITUBE = NLOC(I+3)
105 NFL = NLOC(I+1)
106 ITYPE = NLOC(I+3)
107 NHST = NLOC(I+6)
108 TYPE = TYPE(I+6 - 9)
109 IFEN = .LT. 900 GO TO 110
110 IFENL = .LT. NMAXI GO TO 110
111 IFETUBE .LT. NMAXL GO TO 110
112 IFETYPE .LT. 400 GO TO 110
113 CALL WPBK
114 CALL EXIT
115 100 IC = NLOC(I)
116 GO TO 20
117 120 FORMAT(IN*, * * ERROR IN CONVECTION DATA INPUT TO CONV2 *)
118 WRITE(6,120) NLOC(I)
119 120 FORMAT(5HO, * ERROR IN CONVECTION DATA INPUT TO CONV2 *)
120 IC = 15, IN = 9
121 CALL WPBK
122 CALL EXIT
123 170 ITYPE = LT + TYPE
124 WP = RDATA(LT)
125 CSA = RDATA(LT+1)
126 ISP = CALL DIOEGI(T(NFL),NDATA(NCP),CP)
127 IFIUP = CALL DIOEGI(T(NFL),NDATA(IFPU),FPU)
128 IFET = CALL DIOEGI(T(IFETUBE),NDATA(IFETUBE),FET)
129 WP = 0.0-ABS(RDATA(LT+2)-ITUBE(I))/FWL
130 PR = IFIUP*CP/IC
131 D = 0.0-CSPUP
132 CALL MIOEGI(RDATA(ILOC(I+1),ST)
133 R = T(CP)+IFC*CA(I2)+IFC*ST
134 200 IC = R
135 220 CONTINUE
136 RETURN
137 C
138 C
SUBROUTINE CONV3
DIMENSION MLOC(11,MLOC(11))
DIMENSION NDATA(13)
!
COMMON RA/RAY, RDATA(13)
COMMON TEMP/ T (11)
COMMON SQDD/ G (11)
COMMON SD/ NNA, NNT, NST
!
EQUIVALENCE (RDATA, NDATA)
EQUIVALENCE (NNT, NST)
!
DATA P1 /16.0000/
!
IF (MLOC(11).EQ. 0) GO TO 20
CALL TOPLIN
IF (MLOC(11).EQ. 0) GO TO 20
!
10 FORMAT(* * INCORRECT NUMBER OF ELEMENTS INPUT TO CONV3 FOR
FLOW DATA, IC = 15, TH = * * * *)
!
CALL VKSIC
CALL EXIT
!
20 L2 = MLOC(2)
L7 = MLOC(7)
!
IF (MLOC(11,49).EQ. 0) GO TO 100
CALL TOPLIN
WRITE(6,493) NLOC(11)
!
90 FORMAT(* * INCORRECT NUMBER OF ELEMENTS INPUT TO CONV3 FOR
CONNECTION DATA, IC = 15, TH = * * * )
!
CALL VKSIC
CALL EXIT
!
100 IC = NLOC(11)
DO 220 IC = 1, IC, 4
AG = NLOC(1+1)
NNT = NLOC(+2)
ITUBE = NLOC(+3)
NWU = NLOC(+4)
!
IF (AG .GT. AGF1) GO TO 110
IF (ITUBE .LT. NDATA(11)) GO TO 170
!
110 IC = NLOC(11)
WRITE(6,120) (NLOC()+1,J1,J1)
!
120 FORMAT(40H* * ERROR IN CONNECTION DATA INPUT TO CONV3 * * *
! 3 IF SWGS = 14, 6H AMT = 013.0, 6H ITUBE = 14, 6H NWU = 15)

10. CALL WLHBC
11. CALL EXIT
12. C
13. 170 CALL DIDE(I, MONTACILZ, ETUBE), MONTAH(N), M)
14. 200 GINC3 = MALT
15. 220 CONTINUE
16. C
17. RETURN
18. END
185. 189. 183. 181. 179. 176. 113. 112. 164. 169. 168. 161. 166. 162. 160. 159. 158. 154. FLORAL CALL 153. 152. 150. CALL LINECK(2) *NEW FLORAL 129. CALL LINECK(2) *NEW FLORAL 128. CALL LINECK(2) *NEW FLORAL 127. CALL LINECK(2) *NEW FLORAL 126. CALL LINECK(2) *NEW FLORAL 125. CALL LINECK(2) *NEW FLORAL 124. CALL LINECK(2) *NEW FLORAL 123. CALL LINECK(2) *NEW FLORAL 122. CALL LINECK(2) *NEW FLORAL 121. CALL LINECK(2) *NEW FLORAL 120. CALL LINECK(2) *NEW FLORAL 119. CALL LINECK(2) *NEW FLORAL 118. CALL LINECK(2) *NEW FLORAL 117. CALL LINECK(2) *NEW FLORAL 116. CALL LINECK(2) *NEW FLORAL 115. CALL LINECK(2) *NEW FLORAL 114. CALL LINECK(2) *NEW FLORAL 113. CALL LINECK(2) *NEW FLORAL 112. CALL LINECK(2) *NEW FLORAL 111. CALL LINECK(2) *NEW FLORAL 110. CALL LINECK(2) *NEW FLORAL 109. CALL LINECK(2) *NEW FLORAL 108. CALL LINECK(2) *NEW FLORAL 107. CALL LINECK(2) *NEW FLORAL 106. CALL LINECK(2) *NEW FLORAL 105. CALL LINECK(2) *NEW FLORAL 104. CALL LINECK(2) *NEW FLORAL 103. CALL LINECK(2) *NEW FLORAL 102. CALL LINECK(2) *NEW FLORAL 101. CALL LINECK(2) *NEW FLORAL 100. CALL LINECK(2) *NEW FLORAL 99. CALL LINECK(2) *NEW FLORAL 98. CALL LINECK(2) *NEW FLORAL 97. CALL LINECK(2) *NEW FLORAL 96. CALL LINECK(2) *NEW FLORAL 95. CALL LINECK(2) *NEW FLORAL 94. CALL LINECK(2) *NEW FLORAL 93. CALL LINECK(2) *NEW FLORAL 92. CALL LINECK(2) *NEW FLORAL 91. CALL LINECK(2) *NEW FLORAL 90. CALL LINECK(2) *NEW FLORAL 89. CALL LINECK(2) *NEW FLORAL 88. CALL LINECK(2) *NEW FLORAL 87. CALL LINECK(2) *NEW FLORAL 86. CALL LINECK(2) *NEW FLORAL 85. CALL LINECK(2) *NEW FLORAL 84. CALL LINECK(2) *NEW FLORAL 83. CALL LINECK(2) *NEW FLORAL 82. CALL LINECK(2) *NEW FLORAL 81. CALL LINECK(2) *NEW FLORAL 80. CALL LINECK(2) *NEW FLORAL 79. CALL LINECK(2) *NEW FLORAL 78. CALL LINECK(2) *NEW FLORAL 77. CALL LINECK(2) *NEW FLORAL 76. CALL LINECK(2) *NEW FLORAL 75. CALL LINECK(2) *NEW FLORAL 74. CALL LINECK(2) *NEW FLORAL 73. CALL LINECK(2) *NEW FLORAL 72. CALL LINECK(2) *NEW FLORAL 71. CALL LINECK(2) *NEW FLORAL 70. CALL LINECK(2) *NEW FLORAL 69. CALL LINECK(2) *NEW FLORAL 68. CALL LINECK(2) *NEW FLORAL 67. CALL LINECK(2) *NEW FLORAL 66. CALL LINECK(2) *NEW FLORAL 65. CALL LINECK(2) *NEW FLORAL 64. CALL LINECK(2) *NEW FLORAL 63. CALL LINECK(2) *NEW FLORAL 62. CALL LINECK(2) *NEW FLORAL 61. CALL LINECK(2) *NEW FLORAL 60. CALL LINECK(2) *NEW FLORAL 59. CALL LINECK(2) *NEW FLORAL 58. CALL LINECK(2) *NEW FLORAL 57. CALL LINECK(2) *NEW FLORAL 56. CALL LINECK(2) *NEW FLORAL 55. CALL LINECK(2) *NEW FLORAL 54. CALL LINECK(2) *NEW FLORAL 53. CALL LINECK(2) *NEW FLORAL 52. CALL LINECK(2) *NEW FLORAL 51. CALL LINECK(2) *NEW FLORAL 50. CALL LINECK(2) *NEW FLORAL 49. CALL LINECK(2) *NEW FLORAL 48. CALL LINECK(2) *NEW FLORAL 47. CALL LINECK(2) *NEW FLORAL 46. CALL LINECK(2) *NEW FLORAL 45. CALL LINECK(2) *NEW FLORAL 44. CALL LINECK(2) *NEW FLORAL 43. CALL LINECK(2) *NEW FLORAL 42. CALL LINECK(2) *NEW FLORAL 41. CALL LINECK(2) *NEW FLORAL 40. CALL LINECK(2) *NEW FLORAL 39. CALL LINECK(2) *NEW FLORAL 38. CALL LINECK(2) *NEW FLORAL 37. CALL LINECK(2) *NEW FLORAL 36. CALL LINECK(2) *NEW FLORAL 35. CALL LINECK(2) *NEW FLORAL 34. CALL LINECK(2) *NEW FLORAL 33. CALL LINECK(2) *NEW FLORAL 32. CALL LINECK(2) *NEW FLORAL 31. CALL LINECK(2) *NEW FLORAL 30. CALL LINECK(2) *NEW FLORAL 29. CALL LINECK(2) *NEW FLORAL 28. CALL LINECK(2) *NEW FLORAL 27. CALL LINECK(2) *NEW FLORAL 26. CALL LINECK(2) *NEW FLORAL 25. CALL LINECK(2) *NEW FLORAL 24. CALL LINECK(2) *NEW FLORAL 23. CALL LINECK(2) *NEW FLORAL 22. CALL LINECK(2) *NEW FLORAL 21. CALL LINECK(2) *NEW FLORAL 20. CALL LINECK(2) *NEW FLORAL 19. CALL LINECK(2) *NEW FLORAL 18. CALL LINECK(2) *NEW FLORAL 17. CALL LINECK(2) *NEW FLORAL 16. CALL LINECK(2) *NEW FLORAL 15. CALL LINECK(2) *NEW FLORAL 14. CALL LINECK(2) *NEW FLORAL 13. CALL LINECK(2) *NEW FLORAL 12. CALL LINECK(2) *NEW FLORAL 11. CALL LINECK(2) *NEW FLORAL 10. CALL LINECK(2) *NEW FLORAL 9. CALL LINECK(2) *NEW FLORAL 8. CALL LINECK(2) *NEW FLORAL 7. CALL LINECK(2) *NEW FLORAL 6. CALL LINECK(2) *NEW FLORAL 5. CALL LINECK(2) *NEW FLORAL 4. CALL LINECK(2) *NEW FLORAL 3. CALL LINECK(2) *NEW FLORAL 2. CALL LINECK(2) *NEW FLORAL 1. CALL LINECK(2) *NEW FLORAL
CALL FLORAL
CALL EXIT

C OUTLET PRESSURE INTO COEFFICIENT MATRIX AND RHS
C
C SOLVE FOR Pressures
C
920 PPRN = NEXT(L20)
955 NEXT(L20+P) = PPRN+1/2 + 2
956 NEXT(L20+1) = PPRN
957 NEXT(L20+2) = PPRN
958 IF .NOT. COP GO TO 930
959 CALL TINECK(T2)
NEW
FLOBAL
960 CALL SENOUT(I,I,0,M0ATRII AFTER REDUCTION IS
FLOBAL
962 CALL TINECK(T2)
NEW
FLOBAL
963 CALL GENOUT(EXT(L28*I),I,NEXIT(L28*I),CPRESSURE
30tES')
NEW
FLOBAL
964 CALL TINECK(T2)
NEW
FLOBAL
965 IF (.NOT. COP) GO TO 930
NEW
FLOBAL
967 CALL TINECK(T2)
NEW
FLOBAL
968 CALL TINECK(T2)
NEW
FLOBAL
969 IF(COP) CALL GENOUT(I,ET(L28*I),NCATL28,'0PRESSURES')
NEW
FLOBAL
970 GO TO 1060
**-1
FLOBAL
971 C FLOBAL
972 C FLOBAL
973 C FLOBAL
974 1020 CALL TOPLIN
975 WRITE(6,1050) IATX(I+1)
1040 FORMAT(I1) + Error in solving PressJ/flow Equations for Net
1050 INDE OR 6, 7, 8 + **
976 CALL FLORAL
977 NTH = L25 - 1
NEW
FLOBAL
978 CALL JCAL
979 CALL EXIT
980 C
981 C
982 C UPDATE PRESSURES
983 C
984 DO 1080 J=1,FPRN
985 NPR = NEXT(L28*J)
986 RDATA(L3*NPR) = ETI(L28*J)
987 CONTINUE
988 C
989 C CALCULATE NEW FLOW RATES
990 C
991 DO 1120 J=9,L20,5
992 J = L14 * J
993 NTI = NDATA(K)
994 NFRM = NDATA(K*2)
995 NT0 = NEXT(L25*NFAR)
996 NTB = RCATA(L3*NTB)
997 TEMP = RCATA(L3*NTB) - RCATA(L3*NT0)
998 TEMP = TEMP/WNTB
999 RDATA(L3*NTB) = TEMP
1000 CONTINUE
1000 DO 1120 J=9,L20,5
1001 J = L14 * J
1002 NTI = NDATA(K)
1003 NFRM = NDATA(K*2)
1004 NT0 = NEXT(L25*NFAR)
1005 NTB = RCATA(L3*NTB)
1006 TEMP = RCATA(L3*NTB) - RCATA(L3*NT0)
1007 TEMP = TEMP/WNTB
1008 RDATA(L3*NTB) = TEMP
1009 CONTINUE
1010 IF (.NOT. COP) GO TO 1120
1011 CALL TINECK(T3)
1012 PNFR = RDATA(L3*NFRM) + NPR
1013 PNT0 = RDATA(L3*NT0) + NPR
1014 WRITE(6,1020) M1,NFRM,NT0,REXT(L3+*NFRM),PRNR,PNT0,UNIT,TEMP
1015 IF (.NOT. COP) GO TO 1120
1016 CALL TINECK(T3)
1017...
SUBROUTINE FLOCNI(LLOC,MLOC,MLOC)

LOGICAL LCP

DIMENSION LLOC(1), MLOC(1), NLOC(1)

COMMON T/ I(1)
COMMON W/( I(1))
COMMON ND, NNT, NGT

EQUIVALENCE (NW, W)

IF(LLOC(I) .LT. 0) GO TO 15
CALL TOPLIN
WRITE(6,IO)
FORMAT(83H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW RATES, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

IF(NW = MLOC(I))
IF(NW .LT. I .OR. NW .GT. 65000) GO TO 13
LCP = .TRUE.
CP = W
GO TO 25
LCP = .FALSE.
REP = NW

IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I)
IF(LCP)
CALL DIDEGI
G(ING) = UC
CONTINUE
RETURN

NW = LLOC(I+1)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+2)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+3)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+4)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+5)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+6)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+7)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+8)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+9)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+10)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+11)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+12)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+13)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+14)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

NW = LLOC(I+15)
IF(MLOC(I) .EQ. 0) GO TO 40
CALL TOPLIN
WRITE(6,30)
FORMAT(93H0*k* INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNI FOR
FLOW CONDUCTION DATA, IC = 15, TH *)
CALL WLKBCK
CALL EXIT

RETURN
SUBROUTINE FLOCNZ(LOC,CP,MLOC)

DIMENSION MLOC(1), NLOC(1)

COMMON /EDU/ N, NDA, NFA, MFT, NFT

C EQUIVALENCE (W,W)

C FLOCNZ

IF(LOC(1) .LT. 0) GO TO 20

CALL TOPLIN

WRITE(6,10) MLOC(I)

FORMAT(5H0

ERROR IN FLOW CONDUCTION DATA, IC = 15, TH = *

CALL KLBCK

CALL EXIT

C FLOCNZ

20 IF(LOC(11) .EQ. 0) GO TO 40

CALL TOPLIN

WRITE(6,30) NLOC(I)

FORMAT(60H

INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCNZ FOR FLOW CONDUCTION DATA, IC = 15, TH = *

CALL KLBCK

CALL EXIT

C FLOCNZ

40 NTB = MLOC(1)

IC = MLOC(11)

GO TO 200

NG = MLOC(11) + 1

ITUBE = MLOC(12)

IF(NG .GT. NGT) GO TO 60

IF(ITUBE .LE. NTB) GO TO 100

CALL TOPLIN

WRITE(6,80) (NLOC(I*J), J = 1,2), CP

FORMAT(60H

ERROR IN FLOW CONDUCTION DATA, IC = 15, TH = *

CALL KLBCK

CALL EXIT

C FLOCNZ

60 NW = MLOC(1)

G(NG) = W*CP

CONTINUE

C FLOCNZ

RETURN

END
SUBROUTINE FLMES(L30,NFL)
LOGICAL LR3, LIJ, COP
DIMENSION NDATA(L30)
COMMON ARRAY/NDATA(N)/
COMMON NITEM/NDATA(N)/
COMMON IFDATA/I,
COMMON I/O/FDDATA/
COMMON POINT/N/LNODE
EQUIVALENCE (RDATA,NDATA), (HL,NHL)
DATA MAXI /65000/
C FLUID LUMP LOOP
DO 200 I=1,IC,3
NFL = NDATA(K) ITYPE = NDATA(K*I) NTL = NDATA(K+2) LTYPE = LT + ITYPE*6 - 6 WP = RDATA(LTYPE*I) CSA = RDATA(LTYPE*2) FLL : RDATA(LTYPE*3) MFF = NDATA(LTYPE*9) NHL = NDATA(LTYPE*5) FF = RDATA(LTYPE*6) IF(LPO) CALL DIOEG1(TINFL),NOATA(NRO), R3
IF(LMU) CALL DIOEG1(T(NFL),NDATA(NMU),U) RE = q9/IXu/wP IF(NHL .GT. 0 .AND. NHL .LT. MAXII) CALL OIDEG1(IE,NDATA(NHL),L) IF(RE .GT. 2000.0) GO TO 100 FF = 0.316/SQRT(SRrRF)) IF(MFF .Ea. 0) GO TO 120 CALL DIOEG1(AE,NCATA(MFF),FF) GO TO 160 IF(RE .LT. 4000.0) GO TO 140 FF = 0.2086092052 0.6236 037?5E-T * RE (-0.655956/E-3I)
R = (FFFFCFLLI/(.0.5SIA/P)*HL)-WNTI/E2C
RSum = RSum + R
C IF(4,31) GO TO 200
200 CONTINUE
300 RETURN
SUBROUTINE FLUX(NFLITP,DATANCRV,DTIME)

C DIMENSION DATA(1)

C COMMON /TIMETIM/ T

C EQUIVALENCE (O,N)

C EF(FTIME .GE. TENER) RETURN

C IF(NFLITP .GT. 0) READ(NFLITP) FLYTIM

NFLITP = IABS(NFLITP)

10 READ(NFLITP) FLYTIM

IF(FLYTIM .GE. TTIME) GO TO 10

WRITE(6,20) FTIME

20 IF(NFLITP .GT. 0) READ(NFLITP) DTIME

IF(NFLITP .GT. 0) RETURN

21 LOC = 1

O = DATANCRV

IC = N

DO 20 J=1,IC,2

20 DATANCRV(J) = DATANCRV(J) + DTIME

30 CONTINUE

LOC = LOC + IC + 1

40 CONTINUE

NFLITP = -NFLITP

RETURN

END
SUBROUTINE GENOUT(NOTA,ISTRT,ISTP,NAME)

C

DIMENSION FMT(12), NAME(22)

DIMENSION NOATA(I)

DATA MAll / 13q217I281

DATA FMT(I), FMT(IZ1 / 6H1IlIIP , I

LOGICAL ONE, CKO

BASE = 6HE12.

ASSIGN 32 TO AAA

NOATA(I) = .FALSE.

ENTRY GENI(NDATA,ISTTISST TP,NAME)

BASE = 6HI9,

ASSIGN 95 TO M

CD = .TRUE.

WRITE(4,10) NAME

FORMAT(ZZI6I)

ONE = .FALSE.

IF(ISTRT .E. I .AND. ISTP .EO. 1) ONE = .TRUE.

CASE = 6HEI2.

ASSIGN 95 TO M

CD = .TRUE.

WRITE(4,10) NAME

FORMAT(ZZI6I)

ONE = .FALSE.

IF(ISTRT .E. I .AND. ISTP .EO. 1) ONE = .TRUE.

CASE = 6HEI2.

ASSIGN 95 TO M

CD = .TRUE.

WRITE(4,10) NAME

FORMAT(ZZI6I)

ONE = .FALSE.

IF(ISTRT .E. I .AND. ISTP .EO. 1) ONE = .TRUE.

CASE = 6HEI2.

ASSIGN 95 TO M

CD = .TRUE.

WRITE(4,10) NAME

FORMAT(ZZI6I)

ONE = .FALSE.

IF(ISTRT .E. I .AND. ISTP .EO. 1) ONE = .TRUE.

CASE = 6HEI2.

ASSIGN 95 TO M

CD = .TRUE.
GO WRITE(6,FMT) NOATA(ISTAT) TO RETURN END

GENOUT

GENOUT

GENOUT
**ARRAY DEFINITIONS**

1. DIMENSION M(1), M(2), M(3)
2. DIMENSION N(1), N(2), N(3), N(4), N(5), N(6)
4. DIMENSION UC, UC(1), UC(2), UC(3), UC(4)
5. DIMENSION D(1), D(2), D(3), D(4), D(5), D(6)
6. DIMENSION S(1), S(2), S(3), S(4), S(5), S(6)

**WORD DEFINITIONS**

1. C-ASS - DESCRIPTOR VALUES FOR THE CURRENT FRAME
2. C-NAME - ITEM NAMES AND DIMENSION INFORMATION ON THE ORDER
3. C-DIM - ITEM NAMES AND DIMENSION INFORMATION ON THE ORIGIN
4. C-TRAN - BUFFER FOR READING HISTORY TAPE RECORDS
5. C-STR - ARRAY FOR STORING THE ITEM TYPE INDICES
6. C-TITLE - TYPE NUMBER TO BE PLOTTED
7. C-TYPE - THE ITEM CODE FOR THE RESPECTIVE ITEM NUMBERS
8. C-KEY - ITEM CODE ARRAY
9. C-KEYB - INDEX TO ITEM TYPE IN BUFFER ARRAY
10. C-ORD - ORDER VALUES FOR THE CURRENT FRAME
11. C-TITLEA - GENERAL TITLE FOR EACH FRAME
12. C-TITLEB - TITLE OF 1-ST AND 2-ND ITEMS ON THE CURRENT FRAME
13. C-TITLEC - TITLE OF 3-RD AND 4-TH ITEMS ON THE CURRENT FRAME
14. C-TITLED - THE ITEM PLOTTING SYMBOLOGY AND DESCRIPTIONS
15. C-XY - ARRAY FOR ITEMS TO BE PLOTTED (INCLUDING TIMES)
16. C-YM - THE MAXIMUM ORDINATE VALUES
17. C-YM - THE MINIMUM ORDINATE VALUES
18. C-INIT - INITIALIZATION

**INITIALIZATION**

1. DATA (30)(1.1,2.1) / 2K
2. DATA (30)(1.2,3.2) / 2K
3. DATA (30)(1.3) / 2K
4. DATA (30)(1.4) / 2K
5. DATA (30)(1.5) / 2K
6. DATA (30)(1.6) / 2K
7. DATA (30)(1.7) / 2K
8. DATA (30)(1.8) / 2K
9. DATA (30)(1.9) / 2K
10. DATA (30)(1.10) / 2K
11. DATA (30)(1.11) / 2K
12. DATA (30)(1.12) / 2K
13. DATA (30)(1.13) / 2K
14. DATA (30)(1.14) / 2K
15. DATA (30)(1.15) / 2K
16. DATA (30)(1.16) / 2K
17. DATA (30)(1.17) / 2K
18. DATA (30)(1.18) / 2K
19. DATA (30)(1.19) / 2K
20. DATA (30)(1.20) / 2K
21. DATA (30)(1.21) / 2K
22. DATA (30)(1.22) / 2K
23. DATA (30)(1.23) / 2K
24. DATA (30)(1.24) / 2K
25. DATA (30)(1.25) / 2K
26. DATA (30)(1.26) / 2K
27. DATA (30)(1.27) / 2K
28. DATA (30)(1.28) / 2K
29. DATA (30)(1.29) / 2K
30. DATA (30)(1.30) / 2K
31. DATA (30)(1.31) / 2K
32. DATA (30)(1.32) / 2K
33. DATA (30)(1.33) / 2K
34. DATA (30)(1.34) / 2K
35. DATA (30)(1.35) / 2K
36. DATA (30)(1.36) / 2K
37. DATA (30)(1.37) / 2K
38. DATA (30)(1.38) / 2K
39. DATA (30)(1.39) / 2K
40. DATA (30)(1.40) / 2K
41. DATA (30)(1.41) / 2K
42. DATA (30)(1.42) / 2K
43. DATA (30)(1.43) / 2K
44. DATA (30)(1.44) / 2K
45. DATA (30)(1.45) / 2K
46. DATA (30)(1.46) / 2K
47. DATA (30)(1.47) / 2K
48. DATA (30)(1.48) / 2K
49. DATA (30)(1.49) / 2K
50. DATA (30)(1.50) / 2K
51. DATA (30)(1.51) / 2K
52. DATA (30)(1.52) / 2K
53. DATA (30)(1.53) / 2K
54. DATA (30)(1.54) / 2K
55. DATA (30)(1.55) / 2K
56. DATA (30)(1.56) / 2K
57. DATA (30)(1.57) / 2K
58. DATA (30)(1.58) / 2K
59. DATA (30)(1.59) / 2K
60. DATA (30)(1.60) / 2K
420 IF (I(TYPE(I), .EQ. KEY(J))) GO TO 430
187., 430 J = J+1
188. IF (J .GT. N) GO TO 420
189. C INCORRECT TYPE CODE
190. GEPLOT
191. C 420 WRITE (6,440) J, I(TYPE(I))
192. C ITEM(I) TYPE CODE IN ERRA3
193. GEPLOT
194. GEPLOT
195. GEPLOT
196. C GO TO 540
197. GEPLOT
198. C 460 IF (|KEY(J)| = 0) GO TO 460
199. C INCORRECT ITEM NUMBER
200. GEPLOT
201. C 480 LOCK(I) = -2
202. GEPLOT
203. C WRITE (6,580) J, I(TYPE(I))
204. C ACT = IABS(I(TYPE(I))
205. GEPLOT
206. C WRITE (6,580) J, I(TYPE(I))
207. C FORMAT (499, NYTMS, IF(TYPL = 1, 2, 3) IS OUT OF RANGE)
208. GEPLOT
209. C GO TO 540
210. GEPLOT
211. C 520 LOCK(I) = KEY(J)+IABS(I(TYPE(I))
212. GEPLOT
213. C SAVE FUNCTION TYPE INDEX
214. GEPLOT
215. C 540 IF(SKEL(I) = J)
216. C ITEM(I) IS SKEL(I)
217. GEPLOT
218. C J = J+1
219. GEPLOT
220. C BUMP ITEM NUMBER AND TEST FOR LAST ITEM
221. GEPLOT
222. C IF (1 .LT. ITEM(I)) GO TO 540
223. GEPLOT
224. C START LOADING THE DATA FROM THE HISTORY TAPE
225. GEPLOT
226. C 560 WRITE (6,980)
227. GEPLOT
228. C FORMAT (<&N, NYTMS, COMPOSITIONING AND READING THE DATA TAPE/I>)
229. GEPLOT
230. C POSITION THE HISTORY TAPE
231. GEPLOT
232. C 600 NYTMS = NSIZE/ITEMS
233. GEPLOT
234. C WRITE (6,980)
235. GEPLOT
236. C FORMAT (<&N, NYTMS)
237. GEPLOT
238. C WRITE (6,NVTMS)
239. GEPLOT
240. C 620 IF (MODEL ,.EQ. 1) WRITE (6,980)
241. GEPLOT
242. C MODEL (BUFR(L),L:TE, MWRD)
243. GEPLOT
244. C WRITE (6,980)
245. GEPLOT
246. C MODEL (BUFR(L),L:TE)
247. GEPLOT
248. C IF (BUFR(L) .EQ. 0) IFIN = 1
249. GEPLOT
250. C CHECK FOR REQUESTED FINAL ITEM
251. GEPLOT
252. C IF (FIN = 1) GO TO 700
253. GEPLOT
254. C 700 WRITE (6,980)
255. GEPLOT
256. C MODEL (BUFR(L),L:TE)
257. GEPLOT
258. C IF (ITEM(I) .EQ. .LT. 752)
259. GEPLOT
260. C BUMP THE BV ARRAY SUBSCRIPT
261. GEPLOT
262. GEPLOT
263. GEPLOT
264. GEPLOT
265. GEPLOT
266. GEPLOT
267.
190. NTPTS = NTPTS + 1
200. IF (NTPTS.GT. NTMEAS) STOP
201. GO TO 100
210. 100 CONTINUE
220. M = M + 1, NTPTS = NTPTS + 1
230. NTYMS = NTYMS + 1
240. IF (NTYMS.GE. NTMEAS) STOP
250. GO TO 100
260. DATA (X(1),X(2),X(3)) = (AVERAGE, MAXIMUM, MINIMUM)
270. DATA (Y(1),Y(2),Y(3),Y(4)) = (AVERAGE, MAXIMUM, MINIMUM, TIME)
280. DATA (Z(1),Z(2),Z(3)) = (AVERAGE, MAXIMUM, MINIMUM)
290. DATA (U(1),U(2),U(3)) = (AVERAGE, MAXIMUM, MINIMUM)
300. DATA (V(1),V(2)) = (AVERAGE, MAXIMUM)
310. DATA (W(1),W(2),W(3),W(4)) = (AVERAGE, MAXIMUM, MINIMUM, TIME)
320. DATA (T(1),T(2)) = (AVERAGE, MAXIMUM)
330. DATA (S(1),S(2),S(3)) = (AVERAGE, MAXIMUM, MINIMUM)
340. DATA (R(1),R(2)) = (AVERAGE, MAXIMUM)
350. DATA (Q(1),Q(2)) = (AVERAGE, MAXIMUM)
360. DATA (P(1),P(2)) = (AVERAGE, MAXIMUM)
370. DATA (O(1),O(2)) = (AVERAGE, MAXIMUM)
380. DATA (N(1),N(2)) = (AVERAGE, MAXIMUM)
390. DATA (M(1),M(2)) = (AVERAGE, MAXIMUM)
400. DATA (L(1),L(2)) = (AVERAGE, MAXIMUM)
410. DATA (K(1)) = AVERAGE
420. DATA (J(1)) = AVERAGE
430. DATA (I(1)) = AVERAGE
440. DATA (H(1)) = AVERAGE
450. DATA (G(1)) = AVERAGE
460. DATA (F(1)) = AVERAGE
470. DATA (E(1)) = AVERAGE
480. DATA (D(1)) = AVERAGE
490. DATA (C(1)) = AVERAGE
500. DATA (B(1)) = AVERAGE
510. DATA (A(1)) = AVERAGE
520. DATA (Z(1),Z(2),Z(3)) = (AVERAGE, MAXIMUM, MINIMUM)
530. DATA (Y(1),Y(2),Y(3),Y(4)) = (AVERAGE, MAXIMUM, MINIMUM, TIME)
540. DATA (X(1),X(2),X(3)) = (AVERAGE, MAXIMUM, MINIMUM)
550. DATA (W(1),W(2)) = (AVERAGE, MAXIMUM)
560. DATA (V(1),V(2)) = (AVERAGE, MAXIMUM)
570. DATA (U(1),U(2)) = (AVERAGE, MAXIMUM)
580. DATA (T(1),T(2)) = (AVERAGE, MAXIMUM)
590. DATA (S(1),S(2)) = (AVERAGE, MAXIMUM)
600. DATA (R(1),R(2)) = (AVERAGE, MAXIMUM)
610. DATA (Q(1),Q(2)) = (AVERAGE, MAXIMUM)
620. DATA (P(1),P(2)) = (AVERAGE, MAXIMUM)
630. DATA (O(1),O(2)) = (AVERAGE, MAXIMUM)
640. DATA (N(1),N(2)) = (AVERAGE, MAXIMUM)
650. DATA (M(1),M(2)) = (AVERAGE, MAXIMUM)
660. DATA (L(1),L(2)) = (AVERAGE, MAXIMUM)
670. DATA (K(1)) = AVERAGE
680. DATA (J(1)) = AVERAGE
690. DATA (I(1)) = AVERAGE
700. DATA (H(1)) = AVERAGE
710. DATA (G(1)) = AVERAGE
720. DATA (F(1)) = AVERAGE
730. DATA (E(1)) = AVERAGE
740. DATA (D(1)) = AVERAGE
750. DATA (C(1)) = AVERAGE
760. DATA (B(1)) = AVERAGE
770. DATA (A(1)) = AVERAGE
780. DATA (Z(1),Z(2),Z(3)) = (AVERAGE, MAXIMUM, MINIMUM)
790. DATA (Y(1),Y(2),Y(3),Y(4)) = (AVERAGE, MAXIMUM, MINIMUM, TIME)
800. DATA (X(1),X(2),X(3)) = (AVERAGE, MAXIMUM, MINIMUM)
810. DATA (W(1),W(2)) = (AVERAGE, MAXIMUM)
820. DATA (V(1),V(2)) = (AVERAGE, MAXIMUM)
830. DATA (U(1),U(2)) = (AVERAGE, MAXIMUM)
840. DATA (T(1),T(2)) = (AVERAGE, MAXIMUM)
850. DATA (S(1),S(2)) = (AVERAGE, MAXIMUM)
860. DATA (R(1),R(2)) = (AVERAGE, MAXIMUM)
870. DATA (Q(1),Q(2)) = (AVERAGE, MAXIMUM)
880. DATA (P(1),P(2)) = (AVERAGE, MAXIMUM)
890. DATA (O(1),O(2)) = (AVERAGE, MAXIMUM)
900. DATA (N(1),N(2)) = (AVERAGE, MAXIMUM)
910. DATA (M(1),M(2)) = (AVERAGE, MAXIMUM)
920. DATA (L(1),L(2)) = (AVERAGE, MAXIMUM)
930. DATA (K(1)) = AVERAGE
940. DATA (J(1)) = AVERAGE
950. DATA (I(1)) = AVERAGE
960. DATA (H(1)) = AVERAGE
970. DATA (G(1)) = AVERAGE
980. DATA (F(1)) = AVERAGE
990. DATA (E(1)) = AVERAGE
1000. DATA (D(1)) = AVERAGE
1010. DATA (C(1)) = AVERAGE
1020. DATA (B(1)) = AVERAGE
1030. DATA (A(1)) = AVERAGE
1040. NTMEAS = NTMEAS + 1
1050. STOP
436.

LOAD THE ABSCISSA VALUES

435.

400.

434.

433.

432.

431.

430.

429.

428.

427.

426.

425.

424.

423.

422.

421.

420.

419.

418.

417.

416.

415.

414.

413.

412.

411.

410.

409.

408.

407.

406.

405.

404.

403.

402.

401.

400.

399.

398.

397.

396.

395.

394.

393.

392.

391.

390.

389.

388.

387.

386.

385.

384.

383.

382.

381.

380.

379.

378.

377.

376.

375.

374.

373.

372.

371.

370.

369.

368.

367.

366.

365.

364.

363.

362.

361.

360.

359.

358.

357.

356.

355.

354.

353.

352.

351.

350.

349.

348.

347.

346.

345.

344.

343.

342.

341.

340.

339.

338.

337.

336.

335.

334.

333.

332.

331.

330.

329.

328.

327.

326.

325.

324.

323.

322.

321.

320.

319.

318.

317.

316.

315.

314.

313.

312.

311.

310.

309.

308.

307.

306.

305.

304.

303.

302.

301.

300.

299.

298.

297.

296.

295.

294.

293.

292.

291.

290.

289.

288.

287.

286.

285.

284.

283.

282.

281.

280.

279.

278.

277.

276.

275.

274.

273.

272.

271.

270.

269.

268.

267.

266.

265.

264.

263.

262.

261.

260.

259.

258.

257.

256.

255.

254.

253.

252.

251.

250.

249.

248.

247.

246.

245.

244.

243.

242.

241.

240.

239.

238.

237.

236.

235.

234.

233.

232.

231.

230.

229.

228.

227.

226.

225.

224.

223.

222.

221.

220.

219.

218.

217.

216.

215.

214.

213.

212.

211.

210.

209.

208.

207.

206.

205.

204.

203.

202.

201.

200.

199.

198.

197.

196.

195.

194.

193.

192.

191.

190.

189.

188.

187.

186.

185.

184.

183.

182.

181.

180.

179.

178.

177.

176.

175.

174.

173.

172.

171.

170.

169.

168.

167.

166.

165.

164.

163.

162.

161.

160.

159.

158.

157.

156.

155.

154.

153.

152.

151.

150.

149.

148.

147.

146.

145.

144.

143.

142.

141.

140.

139.

138.

137.

136.

135.

134.

133.

132.

131.

130.

129.

128.

127.

126.

125.

124.

123.

122.

121.

120.

119.

118.

117.

116.

115.

114.

113.

112.

111.

110.

109.

108.

107.

106.

105.

104.

103.

102.

101.

100.

99.

98.

97.

96.

95.

94.

93.

92.

91.

90.

89.

88.

87.

86.

85.

84.

83.

82.

81.

80.

79.

78.

77.

76.

75.

74.

73.

72.

71.

70.

69.

68.

67.

66.

65.

64.

63.

62.

61.

60.

59.

58.

57.

56.

55.

54.

53.

52.

51.

50.

49.

48.

47.

46.

45.

44.

43.

42.

41.

40.

39.

38.

37.

36.

35.

34.

33.

32.

31.

30.

29.

28.

27.

26.

25.

24.

23.

22.

21.

20.

19.

18.

17.

16.

15.

14.

13.

12.

11.

10.

9.

8.

7.

6.

5.

4.

3.

2.

1.
C LOAD THE RESPECTIVE ORDINATE VALUES.

1520 CALL GEPLT524
1525 GO TO 1520

1530 IF(NAVG.NEQ.1) GO TO 1575

1535 TITLE(1) = MVKEY(1)
1540 TITLE(2) = MVKEY(2)
1545 TITLE(3) = MVKEY(3)
1550 DO 1553 = 1,ሪ

1555 TITLE(I) = MVKEY(I)
1560 NPY = 1500  -  10*NC
1565 CALL RATEKEYS, NPV, 1502, 90, 1, MVKEY, I
1570 ISW = 1
1575 DO 1582 = 1,556, NTPTS
1580 ABSL(3) = ABSL(START+1)
1585 NDAT = 1
1590 CALL GEPLT531
1595 CONTINUE

1600 J = J +1
1605 K = K +1

1610 IF(J.LT.ITEMS) GO TO 1560

1615 IF(J.EQ.ITEMS) GO TO 1620

1620 IF(ITEMS.EQ.0) GO TO 1500

1625 EXECUTE THE APPROPRIATE PLOTTING SUBROUTINE CALL GEPLT525

1630 GO TO 1555

1635 C Branch to the appropriate plotting subroutine call GEPLT526

1640 C Error item GEPLT532

1645 C Too many curves GEPLT529

1650 C GEPLT530

1660 C Bump the item and curve counters GEPLT533

1670 1590 IF(NAVG.EQ.0) GO TO 1575

1675 TITLE(1) = MVKEY(1)
1680 TITLE(2) = MVKEY(2)
1685 TITLE(3) = MVKEY(3)
1690 DO 1693 = 1,5

1695 TITLE(I) = MVKEY(I)
1700 NPV = 1000 - 10*NC
1705 CALL RATEKEYS, NPV, 1023, 90, 1, TITLE, I
1710 ISW = 1
1715 DO 1722 = 1,556, NTPTS
1720 ABSL(3) = ABSL(START+1)
1725 NDAT = 1
1730 CALL GEPLT525
1735 CONTINUE

1740 J = J +1
1745 K = K +1

1750 IF(J.LT.ITEMS) GO TO 1700

1755 IF(J.EQ.ITEMS) GO TO 1760

1760 IF(ITEMS.EQ.0) GO TO 1500

1765 EXECUTE THE APPROPRIATE PLOTTING SUBROUTINE CALL GEPLT526

1770 GO TO 1725

1775 C Check for end of items GEPLT541

1780 C GEPLT542
1785 C GEPLT543

1790 C Check for new grid GEPLT544

1795 C GEPLT545

1800 C Reference the new grid set for GEPLT546

1805 C GEPLT547

1810 C GEPLT548

1815 C GEPLT549

1820 C GEPLT550
1620 CONTINUE
128 = 0
140 CONTINUE
64 = 0
145 CONTINUE
64 = 1
149 CONTINUE
60 to 240
154 CONTINUE
END
SUBROUTINE GOPLOT(SYM)

DIMENSION ASYM(50), DSM(50), BUFR(100), ABS1(3), GMOI(1)

COMMON NPTS(1), TPS, BUFR

EQUIVALENCE (BUFR(I), ABS1(I)), (BUFR(200*3), GMOI(I))

INTEGER ISYM(2)

DATA ISYM(1:2, 4, 1, 4, 1): /  

FP = NSYM + 2 + 4  
DT = ABS(NPTS) - ABS1(1)  
NP = FP * DT / TPS + 1.5  
MP = NP - NP1  

Asym(1) = ABS(1)  
Asym(NP) = ABS(NPTS)  
(ASYM(NP)) = ORD(NPTS)  

KK = 0  
DO 10 I = 1, MP  
KK = KK + K  
10 CONTINUE  

DO 15 I = 2, NPTS  

25. IY = NYS(NP)  

CALL LINESV(IX, IY, 1, 1, 1, NSYM, SYR, M)  

15 CONTINUE  

RETURN  

END
SUBROUTINE HEATER(D, QHT, KDE, TSEN, TM, TOFF)

1. C

2. IF(TSEN .LT. TM) GO TO 200

3. IF(TSEN .GT. TOFF) GO TO 100

4. G = G + QHT*KODE

5. RETURN

6. 100 CODE = 0

7. RETURN

8. 200 CODE = 1

9. 300 G = 0 + QHT

10. RETURN

11. END

A-61
SUBROUTINE MSTRY(PR, VP, U, TINC)

 DIMENSION PR(I), VP(I), W(I)

 EQUIVALENCE (NPR, PR(I)), (NVP, VP(I)), (NWM, W(I))

 DATA (I) = 10/
 DATA (LL) = 01/
 DATA (LT) = 21/

 DT = CON(I)
 PAN = PR(I)
 VPN = VP(I)
 WN = W(I)
 IF (E = 0) GO TO 10
 IF (PR(I) .LT. 1) NPR = 1
 IF (NPR .LT. I) NPR = I
 IF (VP(I) .LT. 1) NVP = 1
 IF (WN .LT. 1) NW = 1
 IF (PW .LT. I) PW = 1
 WRITE(LT) HEADER, (LL, I:1, 6), NPR, NVP, LL, LL, NW, LL, NSL
 WRITE(LT) HEADER, (E = 0) CALL NSREA0(I)
 WRITE(LT) HEADER, (E = 0) CALL NSREA0(I)
 T = TIME2 + DT
 IF (CON(I) * 0.000001 .LT. CON(3)) GO TO 19
 CALL HSTTIM(I)
 KK = 0
 GO TO 50
 10 TIME2 = TIME2 + DT
 IF (TINC .LT. 1) TINC = 1
 GO TO 10
 12 CAT = TIME2
 CALL HSTTIM(I)
 IF (CON(I) * 0.000001 .LT. CON(3)) GO TO 19
 CALL HSTTIME(I)
 KE = 0
 END FILE LT
 50 CONTINUE
 RETURN

 SUBROUTINE HSTTIME(I)

 WRITE(LT) ITIME, (PA(I), I:1,NPR), VP(I), (E = 0)
 WRITE(LT) ITIME, (PA(I), I:1,NPR), VP(I), (E = 0)
 RETURN
 END
1. SUBROUTINE HICT (E1, E2, E3, E4, E5, E6, E7, E8, E9) HICT
2. C ANALYSIS OF COUNTER FLOW HEAT EXCHANGERS HICT
3. DIMENSION E1, E2, E3, E4, E5, E6, E7, E8, E9) HICT
4. EQUIVALENCE (E1, E2, E3, E4, E5, E6, E7, E8, E9) HICT
5. C HICT
6. UA = E1 HICT
7. F1(E1) = E2 HICT
8. F1(E2) = E3 HICT
9. E3(F1) = E4 HICT
10. E4(F2) = E5 HICT
11. TIN(1) = E6 HICT
12. TIN(2) = E7 HICT
13. TOUT(1) = E8 HICT
14. TOUT(2) = E9 HICT
15. D2 = TOUT(1) HICT
16. IF(F1(E1).LT.0.0) GO TO 10 HICT
17. CONTINUE HICT
18. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 3 HICT
19. TAVG = 0.5*(TIN(1)+TOUT(1)) HICT
20. CALL OIDEGI(TAVG,1,E1) HICT
21. CONTINUE HICT
22. IF(E3(F2).LT.1. OR. (E3(F2).GT.65000) GO TO 6 HICT
23. TAVG = 0.5*(TIN(2)+TOUT(2)) HICT
24. CALL OIDEGI(TAVG,2,E2) HICT
25. CONTINUE HICT
26. CP1 = CP(E1) HICT
27. IF(LPABS(E1).LE.99999 .AND. (LPABS(E1).GT.0) GO TO 20 HICT
28. CALL D20EGI(E1,1,1) HICT
29. IS = 1 HICT
30. IL = 1 HICT
31. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
32. IS = 2 HICT
33. IL = 1 HICT
34. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
35. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
36. EFF = 1.0 HICT
37. GO TO 50 HICT
38. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
39. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
40. E = (TOUT(1)+TOUT(2)) HICT
41. EFF = (E).LT.0.0) GO TO 40 HICT
42. EFF = (E).LT.0.0) GO TO 40 HICT
43. TOUT(1) = TOUT(2) + (EFF*(TOUT(1)+TOUT(2))) HICT
44. TOUT(2) = TOUT(1) + (EFF*(TOUT(1)+TOUT(2))) HICT
45. X = TOUT(1) HICT
46. X = TOUT(2) HICT
47. RETURN HICT
48. CONTINUE HICT
49. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
50. IF(E3(F1).LT.1. OR. (E3(F1).GT.65000) GO TO 30 HICT
51. CALL WRERR HICT
52. CALL EXIT HICT
53. END HICT
SUBROUTINE MICROS(I1, I2, J, X, S, T, CP, IUA)

C ANALYSIS OF CROSS FLOW HEAT EXCHANGERS
DIMENSION FR(I), CP(I), T(I), TOUT(I), WC(I)
EQUIVALENCE (NCP,CP(I))

C U = I
7. FR(I) = X
8. FR(2) = X
9. CP(I) = X
10. CP(2) = X
11. TIME(I) = X
12. TIME(2) = X
13. TOUT(I) = X
14. TOUT(2) = X
15. GO TO (11, 2)
16. IF(FR(I) .LT. 0.0) GO TO 10
17. CONTINUE
18. IF(NCP(I) .LT. 1. OR. NCP(I) .GT. 65000) GO TO 10
19. TAVG = 0.5*(T(I)*TOUT(I))
20. CALL ODEGI(TAVG, IUA, CP(I))
21. IF(NCP(2) .LT. 1. OR. NCP(2) .GT. 65000) GO TO 6
22. TAVG = 0.5*(T(2)*TOUT(2))
23. CALL ODEGI(TAVG, I, CP(2))
24. CONTINUE
25. WC(I) = FR(I)*CP(I)
26. WC(2) = FR(2)*CP(2)
27. IF(IBS(IUA) .LE. 99999. AND. IABS(IUA) .ST. 0) GO TO 25
28. CALL ODEGI(FR(I), FR(2), X, IUA)
29. IS = I
30. IL = I
31. IF(WCP(I) .LE. UCP(2)) GO TO 20
32. JS = 2
33. JS = I
34. WCPRAT = WCP(I)/WCP(J)
35. EFF = 1.0
36. GO TO 70
37. E = EXP(-UA..78*WCP(I)*2.22/UA(I)*UA(I))
38. EFF = 1. - EXP-WCP(J)*UA(I)*UA(I))
39. GO TO 70
40. UAS = UA/WCP(I)
41. UAL = UA/WCP(J)
42. EFF = UAS/IUA(I).*EXP-UAS31.*UAL/IUA(I).-.1)
43. GO TO 70
44. EFF = (1.-EXP-WCP(I)*UAL/IUA(I)*UAS/IUA(I))
45. GO TO 70
46. EFF = (E-EF*WCP(I)*UAL/IUA(I)*UAS/IUA(I))
47. TOUT(I) = T(I) + EFF*(T(I)-TOUT(I))
48. TOUT(II) = T(I) + EFF*(T(I)-TOUT(I))
49. XR = TOUT(I)
50. XR = TOUT(II)
51. RETURN
52. EXECUTION TERMINATED IN SUBROUTINE MICROS(I1, I2, J, X, S, T, CP, IUA)
SUBROUTINE MIEFF (X, X2, X3, X4, X5, X6, X7, X8, X9)

C ANALYSIS OF HEAT EXCHANGERS WITH EFFECTIVENESS GIVEN

DIMENSION (X(2), X(2), X(2), X(IN), X(IN), X(IN))

EQUIVALENCE (IEFF, EFF), (XCP, CP)

EFF = X1
T(X2) = X3
Y(X3) = X4
Z(X4) = X5
W(X5) = X6
X(X6) = X7
Y(X7) = X8
Z(X8) = X9

DO 10 X = 1, 2
10 CONTINUE

K = (T(X3) * T(X4))
CALL DIPM(EFF, XCP, K)

3 CONTINUE

K = (T(X5) * T(X6))
CALL DIPM(EFF, XCP, K)

1 CONTINUE

CP(I) = FR(I) * CP(I)

IF (IABS(NEFF) .LE. 99999) GO TO 15

IF (IABS(NEFF) .GT. 0)

CALL OIOEG(FR(), FR(2), X, EFF)

15 = -1:
16 = 2

IF (WCP(I) .LE. MCP(2)) GO TO 20

17 = TOUT(I)
18 = TOUT(2)

RETURN

100 MIEFF (X1, X2, X3, X4, X5, X6, X7, X8, X9)
SUBROUTINE HIPAR (UA, FR1, FR2, WCP, IUPRAT, EFF)

C ANALYSIS OF PARALLEL FLOW HEAT EXCHANGERS

DIMENSION CP(2), FR(2), WCP(2), TOUT(2), MCP(2)

EQUIVALENCE (MU, UA), (MCP, CP)

C

UA = 1
FR(1) = 82
FR(2) = 83
CP(1) = 84
CP(2) = 85
TIN(1) = 36
TIN(2) = 17
TOUT(1) = 80
TOUT(2) = 90
DO 10 I = 1, 2
IF(FR(I).LT. 0.0) GO TO 100
10 CONTINUE
IF(NUA(I).LT. 0.0) GO TO 60
1 CONTINUE
CALL DlOEG(TAVG, x4, CP(I))
2 IF(NCP(I).LT. I OR. NCP(I) .GT. 65000) GO TO 60
3 TAVG = 0.5*(TIN(I)*TOUT(I))
4 CALL DIDEGI(TAVG, 15, CP(I))
5 IF(NCP(2).LT. 1 OR. NCP(2) .GT. 65000) GO TO 60
6 TAVG = 0.5*(TIN(2)*TOUT(2))
7 CALL DIDEGI(TAVG, 15, CP(2))
8 CONTINUE
CP(I) = FR(I)*NEW HX
ACP(2) = FR(2)*CP(2)*NEW HX
9 IF(IABSINUA).LE. 99999. AND. IABS(NUA).GT. 0.0) GO TO 30
10 CALL D2OEGI(FR(I), FR(2), XI, UA)
11 IS = 2
12 IL = 2
13 IF(INCP(1).LE. WCP(1)) GO TO 20
14 WCP(1) = WCP(1)+WCP(I)
15 IF(INCP(2).LE. WCP(2)) GO TO 20
16 WCP(2) = WCP(2)+WCP(I)
17 IF(INCP(1).LE. WCP(1)) GO TO 20
18 WCP(1) = WCP(1)+WCP(I)
19 IF(INCP(2).LE. WCP(2)) GO TO 20
20 WCP(2) = WCP(2)+WCP(I)
21 IF(WCP(1).LT. WCP(1)) GO TO 20
22 WCP(1) = WCP(1)+WCP(I)
23 IF(WCP(2).LT. WCP(2)) GO TO 20
24 WCP(2) = WCP(2)+WCP(I)
25 IF(NUA(I).LT. 0.0) GO TO 60
26 NEW HX
27 IF(INCP(I).LE. WCP(I)) GO TO 20
28 WCP(I) = WCP(I)+WCP(I)
29 IF(WCP(I).LT. WCP(I)) GO TO 20
30 WCP(I) = WCP(I)+WCP(I)
31 IF(WCP(I).LT. WCP(I)) GO TO 20
32 WCP(I) = WCP(I)+WCP(I)
33 IF(WCP(I).LT. WCP(I)) GO TO 20
34 WCP(I) = WCP(I)+WCP(I)
35 IF(WCP(I).LT. WCP(I)) GO TO 20
36 WCP(I) = WCP(I)+WCP(I)
37 GO TO 50
38 50 EFF = 1.0-EXP(-IUA/WCP(I))/WCP(I)*UA/WCP(I)*IUPRAT
39 51 TOUT(I) = TIN(I) - EFF*(TIN(I)-TOUT(I))
40 52 TOUT(I) = TIN(I) - WCP(I)*UA/WCP(I)*IUPRAT
41 53 IF(NUA(I).LT. 0.0) GO TO 60
42 54 RETURN
43 60 RETURN
44 100 WRITE(6, 101) FR(I)
45 101 FORMAT(131(1CH' ', THE NEGATIVE FLOW RATES EIS NOT ALLOW
46 RED. EXECUTION TERMINATED IN SUBROUTINE HIPAR//131(1CH'')
47 CALL XBACK
48 CALL EXIT
49 END
SUBROUTINE LINECK(I)
COMMON /FIONC/ N(I)
N(I) = N(I) + 1
L(I) = M(29) + 1
RETURN
END
133. C
134. CALL RISH(III, IV, 3000, 90, 1, 1, 1, MLAST)
135. IF = 1
136. IN = 1
137. IF(97) 1044 GOTO 1100
138. PRINT DECMAL POINT
139. CALL AIK11IV, I, 3000, 90, 1, 1, I, IV, LAST
140. CONTINUE
141. 1065 IF(IY.GT.0) GOTO 1068
142. PRINT TRAILING NUMERALS
143. CALL RITH(III, IV, 3000, 90, 1, 1, MJC, ACDIV, MLAST)
144. PRINT MINUS SIGN
145. CALL RITH(IIY, IV, 3000, 90, 1, 1, M+, MLAST)
146. CONTINUE
147. GO TO 140
148. CONTINUE
149. CONTINUE
150. CALL LWDTH(I, I1TAY )
151. RETURN
152. END
SUBROUTINE MFS0(A,N)
    DIMENSION A(N,N)
    DOUBLE PRECISION D,PIV, OSM
    C
    C INITIALIZE DIAGONAL-LOOP
    C
    PIV = 0
    DO 11 I=1,N
        PIV = PIV + X
    11        END = PIV
    LEND = N - 1
    C
    C START FACTORIZATION-LOOP OVER K-TH ROW
    C
    DO 13 K=2,N
        OSM = 0.00
        IF(K.LEN) 2,4,2
        C
        C START INNER LOOP
        C
        13 DO 15 L=1,LEN
        15        DSUM = DSUM + (PIV - L)
        END = END - L
        DSUM = DSUM + (PIV - L)
        4    IF(LEN) 2,4,2
        C
        C TRANSFORM ELEMENT A(L,L)
        C
        9 OSM = DSUM - OSM
        IF(K-LEN) 10,5,10
        C
        C TEST FOR NEGATIVE PIVOT ELEMENT AND FOR LOSS OF SIGNIFICANCE
        C
        10 IF(OSUM).EQ.1.0,5,12
        C
        C COMPUTE PIVOT ELEMENT
        C
        9 PIV = DSUM(OSUM)
        25 IF(PIV).EQ.0.0,26,27
        C
        C TRANSFORM PIVOT ELEMENT
        C
        26 D = 1.0/PIV
        27 OSM = 0.00
        GOTO 11
4    OSM = OSM + (PIV - L)
        END = END - L
        12 RETURN
73    11 END = END + 1
        RETURN
32    12 RETURN
34    END
 999
I. FUNCTION NBLANK (WORD, N)

2. INTEGER WORD(N), BLANK

3. DATA BLANK/6H

4. M = N + 1

5. DD 20 M-1, N

6. I = M - N

7. IF (WORD(I) - BLANK) 10, 20, 40

8. 20 CONTINUE

9. 40 NBLANK = 6 * I

10. RETURN

11. END
SUBROUTINE NEWTAP(PR,VP,W,TMPTIM)

INTEGER HEADER(12), PR(I), VP(I), M(I)

COMMON /FICOM/ CON(I)

COMMON /STEP/ T(I)

COMMON /DIMENS/ NNA, NNL

COMMON /TIME/ IT(I)

COMMON /IDIMENS/ NNO, NNL, NTL

DATA IUT, I24, I15

READ(IUT) HEADER, (NP,I=1,6), NPR, NVP, NP, M, N, MP, NSL

IF(PR(I) .NE. NPR) GO TO 10
IF(VP(I) .NE. NVP) GO TO 10
IF(M(I) .NE. NW) GO TO 10
IF(NTL .EQ. NSL) GO TO 20
CALL TOPLIN

MWRITE(6,15,AEADER,PR(I),VP(I),W,NTL, NPR, NVP, NP, M, N, NSL)

FORMAT(82HO*ITEM COUNTS FROM HISTORY TAPE DO NOT MATCH ITEM

COUNTS FOR THIS RUN = = = = = = BY 2 IN THE HISTORY TAPE LABEL IS =
1 126 // 126 ITEM COUNTS FOR THIS RUN ARE = = = = = = = = = = = = =
3 3NP, 15, 3NPV, 15, 3NW, 15, 3NSL /
2 24 43 ITEM COUNTS FROM THE HISTORY TAPE ARE = = = = = =
3 3NP, 15, 3NPV, 15, 3NW, 15, 3NSL /

CALL VERB

CALL EXIT

READ(IUT) ITIME, (NP(I+1),I=1,NPR), (VP(I),I=1,NVP),

1 (W(I+1),I=1,NTL), (T(I),I=1,NTL)

IF(IXTIME .LT. FIXTIME) GO TO 30
IF(IXTIME .LT. TMPTIM) GO TO 20
GO TO 50
GO TO 10

WRITE(6,40)

FORMAT(62HOINITIAL TEMPERATURES AND VALVE POSITIONS INPUT FROM U-TAPE AT 212.5

RETURN

END
SUBROUTINE NONLIN

INCLUDE COMM, LIST

NAC = NNA+NND

DO 50 KEE = 1, NAC

50 KEE = 0

KEEP1 = NDI

KEEP2 = NTH + NAC

NTH = NTH + NAC

CALL VARBL1

NDIM = KEEP1

NTH = KEEP2

J1 = 0

J2 = 1

IF(NND.EQ.0) GO TO 86

D3 85 I = 1, NND

INCLUDE VARC, LIST

INCLUDE VARD, LIST

10 J1 = J1+1

LG = FLO(S, 16, NS01(J1))

LTA = FLO(22, 14, NS01(J1))

INCLUDE VARG2, LIST

C CHECK FOR LAST CONDUCTOR

IF(NSO1(J1).GT.0) GO TO 10

85 CONTINUE

86 CONTINUE

IF(LG.EQ.0) GO TO 85

LTA = FLD(22, I4, NSO1(J1))

INCLUDE VARG2, LIST

C CHECK FOR LAST CONDUCTOR

IF(NSO1(J1).GT.0) GO TO 85

165 CONTINUE

166 CONTINUE

RETURN

END
SUBROUTINE NTWRL(T)
!
LOGICAL LP, LFM, LAR, LDP, COP, FIRST
!
DIMENSION RDATA(I)
!
COMMON /ARRAY/ RDATA(I)
COMMON /DATA/ L2, L3, L4, L5, L6, L7, L8, L9
COMMON /DATA/ LP, LAR, LDD, LDU, R1, R2, RMS, R3, R4, G2
COMMON /DATA/ TOL, NPASS, EPS, FROF, ITU
!
COMMON /SPACE/ NOIM, NTH, NEXT(I)
!
EQUIVALENCE (ROAT, RDATA)
!
Cw declare loop
!
DO 500 J = 4, LZ0, 5
!
!
IF (FIRST) GO TO 510
!
!
IF (.NOT. COP) GO TO 550
!
!
CALL LINECK(I)
!
WRITE(6, 960) NTS, NFRM, NT3, KOAT, ROATA(L2*NFR)
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
1. C SUBROUTINE NETR(IN,NIN,MPA,NDT)
2. C LOGICAL LDP, LIFR, LAR, LDP, COP, FIRST
3. C DIMENSION NDATA
4. C COMMON ARRAY / NDATA/
5. COMMON FDATA / L1, L2, L3, L4, L5, L6, L7, L8, L9
6. COMMON FDATA / LDP, LIFR, LAR, LDP
7. COMMON FDATA / COP, L12, N3, N4, N5, N6, N7, N8, N9
8. COMMON FDATA / TL, NMPASS, EPS, FRDF
9. COMMON FSPACE / N1, N1, N1
10. C EQUIVALENCE (DATA,NDATA)
11. C LDP/LDATA1+1)
12. LDP = NTH + 1
13. NEXI(L1) = N1
14. NPPN = 0
15. FIRST = .TRUE.
16. EPDP = 1.0
17. C PASS LOOP
18. DO 940 NMPASS=1,NMPASS
19. DATA = 0.0
20. C IF (NOT. COP) GO TO 470
21. IF (.NOT. FIRST) CALL TOPLIN
22. CALL LINECK(4)
23. WRITE(6,440) NMPASS, NDATA1+1)
24. 440 FORMAT (A12, L19, A19, A10, A19, A10, A19, A19, A10)
25. C TUBE LOOP
26. C 470 DO 520 J=1,120,5
27. K = L1 + J
28. NTH = NDATA(K)
29. NPPN = NDATA(K+1)
30. N3 = NDATA(K+2)
31. N4 = NDATA(K+3)
32. N5 = NDATA(K+4)
33. C IF (FIRST) GO TO 475
34. NPPN = NEXI(L3+NPPN)
35. N3 = NEXI(L3+N3)
36. C 475 IF (.NOT. COP) GO TO 500
37. CALL LINECK(3)
38. WRITE(6,440) NTH, NPPN, N3, KDAT, NDATAL2+NTB)
39. 440 FORMAT (A12, L19, A19, A19, A19, A19, A19)
40. 1 = TL + N3 = 110, 8 = THNAT = 110, 8 = THN. (NTB) = 110. 0.1
41. C 500 IF (KDAT) 405,417,510
42. 405 NTH = NTH + NPPN = 1
43. CALL MURNL(3,NDATAL2+NTB, N, NDATAL3+NTB)
44. NTH = L12 = 1
45. NDATAL4+NTB) = NDATAL2+NTB) (NDATAL3+NDATAL4+NTB)
46. CALL MURNL(3,NDATAL2+NTB) (NDATAL3+NDATAL4+NTB)
CALL LINECK(3, NEW)
CALL NETWRKI(6, I506) NPASS, NDATA(I, LIII) NEW
CALL NETWRKI(63, 906)

FORMAT(I1, 3* *) CONTINUING PASS I5, 13H FOR RETURK A

CALL NETWRKI(64, 910) NEW
CALL FLRES(L30, WTB)

CALL NETWRKI(65, 515)

CALL NETWRKI(66, C)

CALL NETWRKI(68, C)

CALL NETWRKI(69, C)

IF(LAR) RCATA(I, NEW NETWKI)

C INTRKI

CALL FLOBAL(PRNMLI, WIN, NPI, NPO, EFRDF, DCFI)

CALL NETWRKI(7, 520)

CALL NETWRKI(8, IF(06, AT, TOL) GO TO 530 NETWKI)

CALL NETWRKI(9, 525)

J=9, LZ0, 5 CONTINUE NETWKI

K = L11*J NETWKI

MFRM: NDATA(K, I)

DISTA(K+2)

NDATA(K) = NEXT(L25NFRM)

NDATA(K+2) = NEXT(L5NTO)

IF(.NOT. LOP) GO TO 525 NETWKI

C NETWKI

C CALCULATE PRESSURE DROP IN TUBE NETWKI

ITB := NATA(K)

NFRM := NDATA(K, I)

NTO = NDATA(K+2)

ADATA(IL, NTIB) = NATA(K) - NATA(L3+NFRM)

ROATA(L3+NTO) - ROATA(L3+NTO)

C) . 525 CONTINUE NETWKI

RETURN NETWKI

FIRST = .FALSE. NEW NETWKI

EFROF = FROF NEW NETWKI

5q0 CONTINUE NETWKI

CALL TOPLIN NETWKI

WRITE(6, 560) NDATA(I+), PIXPASS, OwmI, TOL NETWKI

560 FORMAT(B85HO NETWKI

FAILED TO CONVERGE TO A SOLUTION FOR PRESSURES FOR NETWORK A6, TH...

2 ox INHMAXIMUM PASSES - 110 NETWKI

3 8I 19HMAXIMUM CHANGE - G13.8 NETWKI

4 81 I9HMAXIMUM ALLOWABLE - G613.8 NETWKI

5 NETWKI

CALL WLKBCK NETWKI

CALL OUTCAL NETWKI

CALL FLPRNT(RDATAIL), ISHFLOW NETWKI

CALL EliT NETWKI

C NETWKI

END NETWKI
1. SUBROUTINE NTURK1(LIN,WIN,MP1,MP3)
2. LOGICAL LV1, LV2, LDL, DLP, COP, FIRST
3. DIMENSION RDATA(1)
4. COMMON SDRATA / RDATA(1) /
5. COMMON SDRATA / L2, L3, L4, L5, L6, L7, L8, L9 /
6. COMMON SDRATA / LV1, LV2, LDL, DLP /
7. COMMON SDRATA / COP, LDL, LDL, R3, R4, RD, RNP, RNP, RNP, ROP /
8. COMMON SDRATA / TOL, MFPA, EPS, FDFF /
9. COMMON SDRATA / NDN, NTP, NEXT(1) /
10. COMMON SDRATA / RDATA(MP1), NTP, NEXT(1) /
11. COMMON SDRATA / RDATA(MP3), NTP, NEXT(1) /
12. EQUIVALENCE (RDATA(MP1), NTP, NEXT(1)) /
13. COMMON SDRATA / RDATA(MP3), NTP, NEXT(1) /
14. COMMON SDRATA / RDATA(MP1), NTP, NEXT(1) /
15. COMMON SDRATA /
16. COMMON SDRATA /
17. L20=ndata(l)*1-1
18. L25 = NTH + 1
19. NEXT(L25) = NDI
20. NFPA = 0
21. FIRST = .TRUE.
22. EPSF = 1.0
23. PASS LOOP
24. D2 = 940 NPASS+1, MIPASS
25. DMAT = 0.0
26. IF (.NOT. COP) GO TO 720
27. IF (.NOT. FIRST) CALL TOPLIN
28. CALL LINECK(4)
29. WRITE (6,rad) NPASS, RDATA(l)+1)
30. 460 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = *)
31. TUBE LOOP
32. 470 D2 = 920 J=1,L20,5
33. 480 X = 1.0 + J
34. 490 NTB = NDATA(X)
35. 500 NFPA = NDATA(X)+1)
36. 510 NDQ = NDATA(X+1)
37. 520 KOT = NDATA(X+1)
38. 530 L3N = NDATA(X+1)
39. IF (FIRST) GO TO 476
40. NFPA = NEXT(L25+NFPA)
41. NTB = NEXT(L25+NTB)
42. IF (.NOT. COP) GO TO 476
43. NDATA(l)=110,13H FOR NETWORK A1, TN = +
44. CALL TOPLIN
45. WRITE (6,rad) NPASS, RDATA(l)+1)
46. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = *)
47. TUBE LOOP
48. 490 D2 = 920 J=1,L20,5
49. 500 X = 1.0 + J
50. 510 NTB = NDATA(X)
51. 520 NFPA = NDATA(X)+1)
52. 530 NDQ = NDATA(X+1)
53. 540 KOT = NDATA(X+1)
55. 560 L3N = NDATA(X+1)
56. IF (.NOT. COP) GO TO 476
57. CALL LINECK(4)
58. WRITE (6,rad) NPASS, RDATA(l)+1)
59. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
60. CALL TOPLIN
61. WRITE (6,rad) NPASS, RDATA(l)+1)
62. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
63. CALL TOPLIN
64. WRITE (6,rad) NPASS, RDATA(l)+1)
65. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
66. CALL TOPLIN
67. WRITE (6,rad) NPASS, RDATA(l)+1)
68. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
69. CALL TOPLIN
70. WRITE (6,rad) NPASS, RDATA(l)+1)
71. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
72. CALL TOPLIN
73. WRITE (6,rad) NPASS, RDATA(l)+1)
74. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
75. CALL TOPLIN
76. WRITE (6,rad) NPASS, RDATA(l)+1)
77. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
78. CALL TOPLIN
79. WRITE (6,rad) NPASS, RDATA(l)+1)
80. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
81. CALL TOPLIN
82. WRITE (6,rad) NPASS, RDATA(l)+1)
83. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
84. CALL TOPLIN
85. WRITE (6,rad) NPASS, RDATA(l)+1)
86. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
87. CALL TOPLIN
88. WRITE (6,rad) NPASS, RDATA(l)+1)
89. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
90. CALL TOPLIN
91. WRITE (6,rad) NPASS, RDATA(l)+1)
92. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
93. CALL TOPLIN
94. WRITE (6,rad) NPASS, RDATA(l)+1)
95. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
96. CALL TOPLIN
97. WRITE (6,rad) NPASS, RDATA(l)+1)
98. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
99. CALL TOPLIN
100. WRITE (6,rad) NPASS, RDATA(l)+1)
101. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
102. CALL TOPLIN
103. WRITE (6,rad) NPASS, RDATA(l)+1)
104. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
105. CALL TOPLIN
106. WRITE (6,rad) NPASS, RDATA(l)+1)
107. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
108. CALL TOPLIN
109. WRITE (6,rad) NPASS, RDATA(l)+1)
110. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
111. CALL TOPLIN
112. WRITE (6,rad) NPASS, RDATA(l)+1)
113. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
114. CALL TOPLIN
115. WRITE (6,rad) NPASS, RDATA(l)+1)
116. 480 FORMAT(I12, * * PASS IS, 12H FOR NETWORK A1, TN = +)
1 * *)
I *)

CALL LIB

CALL ELF

CALL RES(N,T,MTB)

CALL MLI

CT

CALL FLRES(L,9TB,NTh)

CALL TUR

CT

CALL FLPRNT(ROATA(L),15HFLOW CONDUCTORS)

CALL EXIT

END
SUBROUTINE PFCS,LOC,NAME)
LOGICAL LVP, LIFR, LAR, LOP, COP, LPU, LPASS, LPRO
DIMENSION ML2(I),MLC2(I),NAME(I)
DIMENSION NDATA(3), EX(I)
COMMON /ARRAY/ I
COMMON /IFDATA/ L, L2, L4, L5, L6, L7, L8, L9
COMMON /IFDATA/ COP, LR0, NR0, O3, LMU, kMU, I’P, GC2
COMMON /IFDATA/ TOL, IXPASS, EPS, PROF
COMMON /POINTIN/ LNOE
EQUIVALENCE (ROATA,NDATA), (EIT,NEIT)
EQUIVALENCE ICON(1),TLM1,DC2
EQUIVALENCE ITSEN,NSEN), (ITSET,NSET)
DATA MAXI /65000/
LVP .FALSE.
LIFR = .FALSE.
LAR = .FALSE.
LOP = .FALSE.
IF(MLOC(I).GT.8) GO TO 20
CALL TOPLIN
WRITE(6,10) MLOC(I), NAME
10 FORMAT(1H0,* *
INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR PFCS
IFLOW DATA, IC = IS, TH = * * // 84 AND 9463
GO TO 140
20 L2 = MLOC(I)
21 IF(NDATA(L2).LE.0) GO TO 40
22 CALL TOPLIN
23 WRITE(6,30) NDATA(L2), NAME
30 FORMAT(12H0,* *
INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR PFCS
IFLOW RATES, IC = IS, TH = * * // 84 ANO 9463
40 GO TO 140
41 C
40 L3 = MLOC(I)
42 IF(NDATA(L3).LE.0) GO TO 60
43 CALL TOPLIN
44 WRITE(6,50) NDATA(L3), NAME
50 FORMAT(12H0,* *
INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR PFCS
IFLOW CONDUCTORS, IC = IS, TH = * * // 84 AND 9463
60 GO TO 140
61 C
60 L4 = MLOC(I)
61 IF(NDATA(L4).EQ.0) GO TO 80
62 CALL TOPLIN
63 WRITE(6,70) NDATA(L4), NAME
70 FORMAT(12H0,* *
INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR PFCS
IFLOW CONDUCTORS, IC = IS, TH = * * // 84 AND 9463
80 GO TO 140
81 C
00 LS = NLOC(3)
.. IF LS.LT. 11 GO TO 120
.. IF ILOC(1).EQ. 03.GT. LVP = .TRUE.
100 LS = NLOC(6)
.. IF LS.LT. 11 GO TO 120
.. IF ILOC(1).EQ. 03.GT. LVP = .TRUE.
120 LS = NLOC(7)
.. IF ILOC(NHAT111).EQ. 03.GT. LVP = .TRUE.
130 FORMAT'THAT = ** INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR
.. IF LOCTYPDATA = IC = 15, IC = * ** // EX #F# 9563
140 WRITE(6,150) NSDFCN(13), NAME PFCS
150 FORMAT'THAT = ** INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR
.. IF SDFCNDATA = IC = 15, IC = * ** // EX #F# 9563
160 IF(LJ = NLOC(11), NAME PFCS
170 FORMAT'THAT = ** INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS FOR
.. IF SDFCNDATA = IC = 15, IC = * ** // EX #F# 9563
180 IF(LJ = NLOC(11), NAME PFCS
190 IF(LJ = NLOC(11), NAME PFCS
200 IF(LJ = NLOC(11), NAME PFCS
210 IF(LJ = NLOC(11), NAME PFCS
220 IF(LJ = NLOC(11), NAME PFCS
230 IF(LJ = NLOC(11), NAME PFCS
240 IF(LJ = NLOC(11), NAME PFCS
250 IF(LJ = NLOC(11), NAME PFCS
260 IF(LJ = NLOC(11), NAME PFCS
270 IF(LJ = NLOC(11), NAME PFCS
280 IF(LJ = NLOC(11), NAME PFCS
290 IF(LJ = NLOC(11), NAME PFCS
300 IF(LJ = NLOC(11), NAME PFCS
310 IF(LJ = NLOC(11), NAME PFCS
320 IF(LJ = NLOC(11), NAME PFCS
330 IF(LJ = NLOC(11), NAME PFCS
340 IF(LJ = NLOC(11), NAME PFCS
350 IF(LJ = NLOC(11), NAME PFCS
360 IF(LJ = NLOC(11), NAME PFCS
370 IF(LJ = NLOC(11), NAME PFCS
380 IF(LJ = NLOC(11), NAME PFCS
390 IF(LJ = NLOC(11), NAME PFCS
400 IF(LJ = NLOC(11), NAME PFCS
410 IF(LJ = NLOC(11), NAME PFCS
420 IF(LJ = NLOC(11), NAME PFCS
430 IF(LJ = NLOC(11), NAME PFCS
440 IF(LJ = NLOC(11), NAME PFCS
450 IF(LJ = NLOC(11), NAME PFCS
460 IF(LJ = NLOC(11), NAME PFCS
470 IF(LJ = NLOC(11), NAME PFCS
480 IF(LJ = NLOC(11), NAME PFCS
490 IF(LJ = NLOC(11), NAME PFCS
500 IF(LJ = NLOC(11), NAME PFCS
510 IF(LJ = NLOC(11), NAME PFCS
520 IF(LJ = NLOC(11), NAME PFCS
530 IF(LJ = NLOC(11), NAME PFCS
540 IF(LJ = NLOC(11), NAME PFCS
550 IF(LJ = NLOC(11), NAME PFCS
560 IF(LJ = NLOC(11), NAME PFCS
570 IF(LJ = NLOC(11), NAME PFCS
580 IF(LJ = NLOC(11), NAME PFCS
590 IF(LJ = NLOC(11), NAME PFCS
600 IF(LJ = NLOC(11), NAME PFCS
610 IF(LJ = NLOC(11), NAME PFCS
620 IF(LJ = NLOC(11), NAME PFCS
630 IF(LJ = NLOC(11), NAME PFCS
640 IF(LJ = NLOC(11), NAME PFCS
650 IF(LJ = NLOC(11), NAME PFCS
660 IF(LJ = NLOC(11), NAME PFCS
670 IF(LJ = NLOC(11), NAME PFCS
680 IF(LJ = NLOC(11), NAME PFCS
690 IF(LJ = NLOC(11), NAME PFCS
700 IF(LJ = NLOC(11), NAME PFCS
710 IF(LJ = NLOC(11), NAME PFCS
720 IF(LJ = NLOC(11), NAME PFCS
730 IF(LJ = NLOC(11), NAME PFCS
740 IF(LJ = NLOC(11), NAME PFCS
750 IF(LJ = NLOC(11), NAME PFCS
760 IF(LJ = NLOC(11), NAME PFCS
770 IF(LJ = NLOC(11), NAME PFCS
780 IF(LJ = NLOC(11), NAME PFCS
790 IF(LJ = NLOC(11), NAME PFCS
800 IF(LJ = NLOC(11), NAME PFCS
810 IF(LJ = NLOC(11), NAME PFCS
820 IF(LJ = NLOC(11), NAME PFCS
830 IF(LJ = NLOC(11), NAME PFCS
840 IF(LJ = NLOC(11), NAME PFCS
850 IF(LJ = NLOC(11), NAME PFCS
860 IF(LJ = NLOC(11), NAME PFCS
870 IF(LJ = NLOC(11), NAME PFCS
880 IF(LJ = NLOC(11), NAME PFCS
890 IF(LJ = NLOC(11), NAME PFCS
900 IF(LJ = NLOC(11), NAME PFCS
910 IF(LJ = NLOC(11), NAME PFCS
920 IF(LJ = NLOC(11), NAME PFCS
930 IF(LJ = NLOC(11), NAME PFCS
940 IF(LJ = NLOC(11), NAME PFCS
950 IF(LJ = NLOC(11), NAME PFCS
960 IF(LJ = NLOC(11), NAME PFCS
970 IF(LJ = NLOC(11), NAME PFCS
980 IF(LJ = NLOC(11), NAME PFCS
990 IF(LJ = NLOC(11), NAME PFCS
A84
GO TO 320
310 LRD = .FALSE.
320 AQ = RDATA(L12+2)
330 MNU = RDATA(L12+3)
340 IF(MNU .LT. 10 .OR. MNU .GT. MAX) GO TO 330
350 IFM = .TRUE.
360 MNU = RDATA(L12+3)
370 GO TO 340
380 IFM = .FALSE.
390 MNU = RDATA(L12+3)
400 GO TO 340
410 CALL EXIT PFCS
C SOLUTION PARAMETERS
C TOL = RDATA(L13+1)
420 MSTOP = RDATA(L13+2)
430 EPS = RDATA(L13+3)
440 PROF = RDATA(L13+4)
450 C VALVES
460 IF(NSENS .GT. 0 .AND. NSENS
470 IC
480 IF(ABS(TSEN-TSET) - L11*150, NSENS .GT. NDATA(L1*8)
490 RDATA(L11*6)
500 IF(NDATA(L10) .LT.
510 CALL WRACK
520 CALL EXIT
C PFCS
530 L4O = RDATA(L15)
540 GO TO 510
550 L4O = RDATA(L15+3)
560 NV = RDATA(L14+1)
570 MNOTE = RDATA(L14+9)
580 IF(MNOTE .EQ. 0) GO TO 580
590 IF(NSENS .GT. 0) GO TO 393
600 IF(NSENS .LT. 0) GO TO 580
610 XSENS = RDATA(L14+8)
620 IC = RDATA(L14)
630 IF(ABS(L1) .LT. 20) GO TO 450
640 IF(ABS(L1) .LT. 20) GO TO 450
650 BSET = RDATA(L14+9)
660 IF(NSENS .GT. 0 .AND. NSENS .LT. 10000) TSEN = XSENS
670 IXSSET = RDATA(L15+7)
680 CALL WRACK
690 CALL EXIT
C VALVES
750 C PFCS
760 CALL TOPLIN
770 WRITE(6,350) IC, NAME
780 IF(NSENS .GT. 10) GO TO 350
790 IF(NSENS .GT. 10) GO TO 350
800 IF(ABS(TSEN-TSET) - L11*150, NSENS .GT. NDATA(L1*8)
810 RDATA(L11*6)
820 GO TO 510
C RATE LIMITED
C PFCS
180go to 390
181 IFABS(TSEN-TSET) - L11*150, NSENS .GT. NDATA(L1*8)
182 IFABS(TSEN-TSET) - L11*150, NSENS .GT. NDATA(L1*8)
183 XCOEFF = MAX(REAL(L15+11*(7*SENS-TSET-TBS1-RDATA(L1*12)+1))
184 RDATA(L15+NV) = MAX(REAL(L15+NV)*XCOEFF, F*L11)
185 GO TO 510
**Error in Document**: The text appears to be a mix of programming code and mathematical expressions, possibly related to a scientific or engineering context. However, the text is not clearly legible due to the quality of the image. The text contains variables, mathematical operations, and what seems to be parts of a program, but the exact content and context are unclear from the provided image.
1400 IFILPASS) GO TO 640
1410 WRITE(6,630) EPS, PROF, EPS
1420 IFILPASS) GO TO 640
1430 WRITE(6,610) EPS, PROF, EPS
1440 EPS = EPS(100) EPS(100)
1450 EPS = EPS(100) EPS(100)
1460 EPS = EPS(100) EPS(100)
1470 EPS = EPS(100) EPS(100)
1480 EPS = EPS(100) EPS(100)
1490 EPS = EPS(100) EPS(100)
1500 EPS = EPS(100) EPS(100)
1510 EPS = EPS(100) EPS(100)
1520 EPS = EPS(100) EPS(100)
1530 EPS = EPS(100) EPS(100)
1540 EPS = EPS(100) EPS(100)
1550 EPS = EPS(100) EPS(100)
1560 EPS = EPS(100) EPS(100)
1570 EPS = EPS(100) EPS(100)
1580 EPS = EPS(100) EPS(100)
1590 EPS = EPS(100) EPS(100)
1600 EPS = EPS(100) EPS(100)
1610 EPS = EPS(100) EPS(100)
1620 EPS = EPS(100) EPS(100)
1630 EPS = EPS(100) EPS(100)
1640 EPS = EPS(100) EPS(100)
1650 EPS = EPS(100) EPS(100)
1660 EPS = EPS(100) EPS(100)
1670 EPS = EPS(100) EPS(100)
1680 EPS = EPS(100) EPS(100)
1690 EPS = EPS(100) EPS(100)
1700 EPS = EPS(100) EPS(100)
1710 EPS = EPS(100) EPS(100)
1720 EPS = EPS(100) EPS(100)
1730 EPS = EPS(100) EPS(100)
1740 EPS = EPS(100) EPS(100)
1750 EPS = EPS(100) EPS(100)
1760 EPS = EPS(100) EPS(100)
1770 EPS = EPS(100) EPS(100)
1780 EPS = EPS(100) EPS(100)
1790 EPS = EPS(100) EPS(100)
1800 EPS = EPS(100) EPS(100)
1810 EPS = EPS(100) EPS(100)
1820 EPS = EPS(100) EPS(100)
1830 EPS = EPS(100) EPS(100)
1840 EPS = EPS(100) EPS(100)
1850 EPS = EPS(100) EPS(100)
1860 EPS = EPS(100) EPS(100)
1870 EPS = EPS(100) EPS(100)
1880 EPS = EPS(100) EPS(100)
1890 EPS = EPS(100) EPS(100)
1900 EPS = EPS(100) EPS(100)
1910 EPS = EPS(100) EPS(100)
1920 EPS = EPS(100) EPS(100)
1930 EPS = EPS(100) EPS(100)
1940 EPS = EPS(100) EPS(100)
1950 EPS = EPS(100) EPS(100)
1960 EPS = EPS(100) EPS(100)
1970 EPS = EPS(100) EPS(100)
1980 EPS = EPS(100) EPS(100)
1990 EPS = EPS(100) EPS(100)
2000 EPS = EPS(100) EPS(100)
2010 EPS = EPS(100) EPS(100)
2020 EPS = EPS(100) EPS(100)
2030 EPS = EPS(100) EPS(100)
2040 EPS = EPS(100) EPS(100)
2050 EPS = EPS(100) EPS(100)
2060 EPS = EPS(100) EPS(100)
2070 EPS = EPS(100) EPS(100)
2080 EPS = EPS(100) EPS(100)
2090 EPS = EPS(100) EPS(100)
2100 EPS = EPS(100) EPS(100)
2110 EPS = EPS(100) EPS(100)
2120 EPS = EPS(100) EPS(100)
2130 EPS = EPS(100) EPS(100)
2140 EPS = EPS(100) EPS(100)
2150 EPS = EPS(100) EPS(100)
2160 EPS = EPS(100) EPS(100)
2170 EPS = EPS(100) EPS(100)
2180 EPS = EPS(100) EPS(100)
2190 EPS = EPS(100) EPS(100)
2200 EPS = EPS(100) EPS(100)
2210 EPS = EPS(100) EPS(100)
2220 EPS = EPS(100) EPS(100)
2230 EPS = EPS(100) EPS(100)
2240 EPS = EPS(100) EPS(100)
2250 EPS = EPS(100) EPS(100)
2260 EPS = EPS(100) EPS(100)
2270 EPS = EPS(100) EPS(100)
2280 EPS = EPS(100) EPS(100)
2290 EPS = EPS(100) EPS(100)
2300 EPS = EPS(100) EPS(100)
2310 EPS = EPS(100) EPS(100)
2320 EPS = EPS(100) EPS(100)
2330 EPS = EPS(100) EPS(100)
2340 EPS = EPS(100) EPS(100)
2350 EPS = EPS(100) EPS(100)
2360 EPS = EPS(100) EPS(100)
2370 EPS = EPS(100) EPS(100)
2380 EPS = EPS(100) EPS(100)
2390 EPS = EPS(100) EPS(100)
2400 EPS = EPS(100) EPS(100)
2410 EPS = EPS(100) EPS(100)
2420 EPS = EPS(100) EPS(100)
2430 EPS = EPS(100) EPS(100)
2440 EPS = EPS(100) EPS(100)
2450 EPS = EPS(100) EPS(100)
2460 EPS = EPS(100) EPS(100)
2470 EPS = EPS(100) EPS(100)
2480 EPS = EPS(100) EPS(100)
2490 EPS = EPS(100) EPS(100)
2500 EPS = EPS(100) EPS(100)
2510 EPS = EPS(100) EPS(100)
A00 = AD
AII = A1 - OPS/WS
YNEW = WS
IF(LPASS).GO TO 820
TEMP = (OPS-DPL)/WS-WL)
A00 = A0 - DPL * TEMP-WL
AII = A1 - TEMP
820 D3 860 J=1,100
FNEW = A00 + YNEW(AII + YNEW(AII + YNEW(AII + YNEW(AII)))
IFABS(FNEW) = CHECK 940,940,940
840 FP = AII + YNEW(2.0*A2 + YNEW(3.0*A3 + YNEW(4.0*A4))
YNEW = NEW - FNEW/FP
845. 860 CONTINUE
850. C
860 CALL TOPLIN
870 WRITE(6,160) NAME
880 WRITE(6,880) NAME
890 FORMAT(II/INSYSTEM TOTAL PRESSURE DROP IS SUPPLIED BY A FOURTH-
891 INGER POLYNOMIAL FUNCTION OF FLOW RATE)
900 CALL WLKBCK
910 CALL EXIT
920. C
930 IF(ABS(WK-WNEW) - TEST) 1000,1000,950
940 DPL(A4+DPI) = YNEW
950 DPL = YMK
960. 980 CONTINUE
970. C
980 CALL TOPLIN
990 WRITE(6,980) NAME
999 FORMAT(II/* *SUBROUTINE PFCS FAILED TO CONVERGE TO A SOLUTION*
999 IN TO TRUE SYSTEM CHARACTERISTICS AND TRUE PUMP CURVE */ */ */
999 Z BE ylabel 9AS)
1000 CALL WLKBCK
1010 CALL INITIAL
1020 CALL EXIT
1030. C
1040 1000 RETURN
1050 END
1. FUNCTION POL(LOC, X)
2. C
3. COMMON /ARRAY/ NDATA, LOC
4. C
5. CALL DIODEG(LOC, NDATA LOC, X)
6. POL = F
7. RETURN
8. END
SUBROUTINE PAN(LOC,N,MODE)  

DIMENSION LOC(I)  

IF (LOC(I).LT.1) GO TO 20  

DO 10 J=1,N  

IF (LOC(J+1).EQ. NODE) GO TO 30  

10 CONTINUE  

DO N = N + 1  

IF (LOC(I).LT.1) GO TO 40  

LOC(I)+1 = NODE  

NODE = N  

RETURN  

30 NODE = J  

RETURN  

40 NEED = N - LOC(I) 

CALL TOPLIN  

WRITE(6,30) NEED  

30 FORMAT(3H0 NEED = N - LOC(I))  

CALL PRLINE  

PRINT 8X, 3HINSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR FLW  

8X FORMAT(3HINSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR FLW)  

CALL WRLINE  

RETURN  

C END
SUBROUTINE PSOR(A,M,N,R,EPST,W)

C SUBROUTINE PSOR SOLVES A SYSTEM OF SIMULTANEOUS EQUATIONS USING A
C STATIONARY POINT ITERATIVE SUCCESSIVE OVERRELAXATION METHOD.

C
C DIMENSION A(M,N), R(M,N), W(N)

C BASE=100+M

C W1 = W - 1.0

C DO 100 K = 1, MAX

C TEMP = W + (R(I)-SUM)/A(I,I) - W1*X(I)

C IF(ABS(TEMP-X(I)) .GT. EPS) GO TO 700

C WRITE(6,6301) MAXK,BIGC,EPS

C RETURN

C 100 CONTINUE

C IF(BIGC-EPS)700,700

C CONTINUE

C CALL CENTR(A,M,N,'COEFFICIENTS OF P(J)')

C CALL CENTR(R,M,N,'RIGHT HAND SIDE')

C CALL CENTR(X,N,N,'COMPUTED VALUES OF P AFTER MAXIMUM ITERATIONS')

C RETURN

C 700 CONTINUE

C 710 FORMAT(I///1X,IP,' CONVERGED TO A SOLUTION FORpressures in MPa,'15
C 720 FORMAT(I///1X,IP,'ITERATIONS')

C RETURN

C END
SUBROUTINE RADIR(NLOC, SIGMA, TIER)

CALCULATION FOR IN CROSS RADIATION

LOGICAL EXPLICIT

COMMON /SOURCE/ G(1)
COMMON /SPACKE/ NOIM, NTH, EX1(1)
COMMON /ARRAY/ CURVO(1)
COMMON /DIM3/ NTH, MAN
COMMON /TEMP/ T(E)
COMMON /ECON/ COM(1)

DIMENSION MLOC(11)
DIMENSION CURVO(1)
EQUIVALENCE (G,EX1(1))
EQUIVALENCE (CON(1), TIME)
EQUIVALENCE (CURVO, NCURVO)
CALL TOPLIN
CALL EXIT

IF(DTIME.EQ.0.0) THEN
   NCURVO(I) = 0
   FORMAT(SHAIN, INCORRECT NUMBER OF ELEMENTS INPUT TO RADIR)
   CALL LKBCK
   CALL EXIT
   NCURVO(I) = NLOC(I)
   ISNA = NLOC(2)
   ISREF = NLOC(3)
   ISCON = NLOC(5)
   NNA = NLOC(6)
   ISNA = NLOC(I)
   IF(INCURVO(I).GT.150,500,500)
   IBEG = ISNA*1
   IEND = ISNA*NCURVO(I)
   CALL TOPLIN
   WRITE(6,900) (NCURVO(KK),KK=IBEG,IEND)
   FORMAT(D'CROSS RADIATION DATA///6XSURFACE DATA///SURFACE NUMBER///SURFACE AREA///NUMBER OF SURFACES=16/111/SURFACE NUMBER=11/SURFACE AREA=11/NUMBER OF SURFACES=11)
   WRITE(6,400) (CURVO(I), I=1,NS)
   IF(NS .LT. NCURVO(ISNA)) GO TO 500
   goto 501
10 IF (CURVO(LOC).LT.1.0) GO TO 10
   CLRVO(LOC) = 0.00001
   CALL EXIT
   NCURVO(LOC) = 1
   IF (CURVO(LOC).LT.1.0) GO TO 20

I. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

II. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

III. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

IV. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

V. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

VI. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

VII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

VIII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

IX. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

X. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XI. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XIII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XIV. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XV. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XVI. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XVII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XVIII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XIX. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XX. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXI. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXIII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXIV. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXV. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXVI. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXVII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXVIII. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXIX. SUBROUTINE RADIR(NLOC, SIGMA, TIER)

XXX. SUBROUTINE RADIR(NLOC, SIGMA, TIER)
110 CONTINUE
120 CALL SYMPK (CURVO, ISEM, IJS, NS)
130 CALL SYMPK (CURVO, ISEM, IJS, NS)
140 CALL SYMPK (CURVO, ISEM, IJS, NS)
150 DO 170 I = 1, NS
160 LOC = ISEM + I - 1
170 CURVOLOC1 = 0.0
180 CONTINUE
190 LOC = ISEM + I - 1
200 CURVOLOC1 = 0.0
210 CONTINUE
SUBROUTINE SIGRACURVOLOCI

190 CONTINUE
191 LDC = LDC + 1
192 DO 211 I = 1, N5
193 LOC = ISMA + 3 * I
194 AREAS = CURVOLC3 + I
195 N5 = CURVOLC3 + 10
196 LOC = ISMA + 1 - I
197 RA = CURVOLC3 + N5
198 FAT3 = CURVOLC3 + 2
199 DO 209 J = 1, N
200 LOC = LOC + 1
201 NODE = NCUAVLC3 + LOC
202 LOC = LOC * 1
203 ARAT = CURVOLC3 + AREAS
204 TP = TIP/NODE - TIER
205 SIGMAT = SIGMAT + TP + TP
206 IR0 = CP/NODE = 0.0 + SIGMAT + ARAT
207 Q(NODE) = Q(NODE) + ARAT(FAT3 + TP + SIGMAT)

200 CONTINUE
201 LOC = LOC * 1
202 300 CONTINUE
203 DO 300 I = 1, N5
204 ISMA = NCUAVLC3 + I - 13
205 TISMA = TISMA + 0.25 + TIER
206 300 CONTINUE
207 RETURN

500 WRITE(6,4601)

600 FORMATTING 1311H+Y/1X; THE SURFACE EMITTIVITY DATA DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
601 FORMAT(1311H+Y/1X; THE SURFACE DATA DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
602 FORMAT(1311H+Y/1X; THE SURFACE REFLECTIVITY DATA DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
603 FORMAT(1311H+Y/1X; THE DATA FOR SURFACE '15,' DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
604 FORMAT(1311H+Y/1X; THE DATA FOR SURFACE '15,' DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
605 FORMAT(1311H+Y/1X; THE DATA FOR SURFACE '15,' DOES NOT HAVE THE CORRECT NUMBER OF VALUES$/1X 1311H+13
606 FORMAT(1311H+Y/1X; THE EXTRA SPACE ARRAY DOES NOT HAVE THE CORRECT NUMBER OF SPACES$/1X 1311H+13
607 FORMAT(1311H+Y/1X; THE SURFACE NUMBER '15,' IN THE SURFACE CONNECT DATA WAS NOT SUPPLIED IN THE SURFACE DATA$/1X 1311H+13
608 WRITE(6,4911)

910 FORMAT(' EXECUTION TERMINATED BY A PROGRAMMED ERROR.')
920 CALL WALKER
930 CALL EXIT
END
& FORMATTED GRID

*10 FORMAT///GRID DATA//

*10 DATA GRID

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE

*10 FORMAT///GRID DATA//

*10 CONTINUE
CVRVO(L) + CVRVO(L) + FA(CVRVOLOC2+NE)  

180 CONTINUE
181 LOC = 4
182 DO 223 I=1,NS
183 LOC = LOC + 1
184 AREAS = CVRVO(L)
185 NN = CVRVOLOC1+1
186 SQ = CVRVOEEN+1-13
187 DO 210 J=1,NN
188 LOC = LOC + 1
189 NODE = CVRVO(L)
190 LOC = LOC + 1
191 NDE = CVRVO(L)
192 LOC = LOC + 1
193 CONTINUE = (NDEE = NODE + 1) + CVRVOLOC + NN + SQ

210 CONTINUE
220 LOC = LOC + 1
221 CONTINUE
222 RETURN
223 FORMAT(IHD)
224 FORMAT(IHD)
225 FORMAT(IHD)
226 FORMAT(IHD)
227 FORMAT(IHD)
228 FORMAT(IHD)
229 FORMAT(IHD)
230 FORMAT(IHD)
231 FORMAT(IHD)
232 FORMAT(IHD)
233 FORMAT(IHD)
234 FORMAT(IHD)
235 CONTINUE
236 CALL 
237 CALL EXIT
238 END
SUBROUTINE REVPOLY I( A, R)

DIMENSION A(I)

EQUIVALENCE (O, N)

D = A(I)
N = N

IF(PODNI,2) .GT. 0) GO TO 20
IF(PODOI,2) .GT. 0) GO TO 16

X = A(2)

IF(Y .GE. A(3)) RETURN

X = A(M)

IF(Y .LE. A(M+11)) RETURN

03 IS I+4,N,2

IF(PODNI,2) IS,10,9

IF(PODNI,2) IS,10,9

IF(Y .LE. A(N-2)) RETURN

RETURN

10 X = A(I)

RETURN

IF(Y .LE. A(N-2)) RETURN

RETURN

15 CONTINUE

GO TO 20

X = A(I)

RETURN

IF(Y .LE. A(M+11)) RETURN

RETURN

RETURN

RETURN

CONTINUE

RETURN

RETURN

RETURN

RETURN

RETURN

CONTINUE

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

RETURN

REVPOL
```c
SUBROUTINE SVASOLIA,U,($) SYMSOL
2.
C
3.
DIMENSION A(I), B5()
4.
DOUBLE PRECISION OIN, IaOR
5.
C
6.
FACTORIZE GIVEN MATRIX BY MEANS OF SUBROUTINE WNSO SYnSOL
7.
C
8.
A: TRANSPOSE(T)vT
9.
C
10.
CALL WNSO(A,NSiS)
11.
C
12.
INVERT UPPER TRIANGULAR MATRIX TPREPARE INVERSION LOOP
13.
C
14.
PIV = N*(N-1)/2
15.
IND = PIV
16.
C
17.
INITIALIZE INVERSION-LOOP
18.
C
19.
J = IND
20.
C
21.
INITIALIZE ROW-LOOP
22.
C
23.
WORK = 0.00
24.
MIN = MIN - 1
25.
LHOR = IPIV
26.
LVER = J
27.
C
28.
START INNER LOOP
29.
C
30.
3 WORK = WORK + LHOR + L(LHOR)
31.
AL2J = -WORK/DIN
32.
J = J - MIN
33.
C
34.
IND = IND - 1
35.
LD = N*(N+1)/2
36.
DO 12 I=1,N,-1
37.
LD = LD - 1
38.
SUM = 0.0
39.
DO 11 J=1,N
40.
SUM = SUM + (LD+J)*B(J)
41.
11 CONTINUE
42.
BE(J) = SUM
43.
12 CONTINUE
44.
C
45.
DO 8 I=1,N
46.
LD = LD + 1
47.
K = LD
48.
C
49.
DO 7 J=1,M
50.
SUM = SUM + A(K)*B(J)
51.
K = K + J
52.
7 CONTINUE
53.
BE(J) = SUM
54.
C
55.
RETURN
56.
RETURN
57.
END
```
SUBROUTINE TIMECK(RTIME, RTIME)

COMMON / FTIME / CON(1)

EQUIVALENCE (CON(1), TIME), (CON(2), TIMEND), (CON(10), OUTPUT)

DATA CON/1.0 /, FTIME = 0.0 /, return

10. CALL CLOCKS(TIME)
20. GO TO 100

100 CALL CLOCKS(TIME)
200 IF(KODE .LE. 0.) GO TO 350

30. WRITE(6, 305) ETIME
40. FORMAT(16H0COMPUTER TIME = F9.3, ON MINUTES)
50. RETURN
60. END

... (code continues)
SUBROUTINE TOPLIN

C

COMMON /TITLE / M(20)

COMMON / FILE / N(10)

C

IF(M(20).EQ. 111) RETURN

M(28) = 11

M(29) = M(29) + 1

WRITE(6,100) M(29), M

100 FORMAT('100 FORTRAN SYSTEMS IMPROVED NUMERICAL DIFFERENCING ANALYZER', $)$

* - * - SINGA - - UNIVAC-1108 FORTRAN-V VERSION PA

*GE 15 // 52 2048

RETURN

**=1

END
SUBROUTINE TPANT

LOGICAL LSRT, CHK

DIMENSION EXT(I)

COMMON /TEMP / T(I)
COMMON /SPACE/ NOIM, NTH, NEXT(I)
COMMON /INDEX/ KON(I)
COMMON /DIPS/ NUM, NAM, NAT
COMMON /INDEX/ LNODE

C EQUIVALENCE (NEXT,EXT)

DATA LSRT / .FALSE. /
DATA NAT / INT /

IF(LN3DE .EQ. 0) CALL NNREAD() CALL STNORD

IF(LSRT) GO TO 50
LSRT = .TRUE.

N0IM = NDIM * NNT
FINDIM .LT. 0) GO TO 100

NNODE = NOIM * NTH
NEXT(NNODE+*N4)

IF(NEIT(LNODE*NN).LE. NEXT(LNODE*NN)) GO TO 20
CHK : .FALSE.
NEXT(NODE#NNI)

CONTINUE

IF(NDI .LT. 12) GO TO 100

J = I
L = 6
M = NTH + 1
IF( .LT. L) M = NEXT(NODE+N)
NEXT(NODE+N) = N1
NEXT(NODE+N1) = NW
CONTINUE

IF (CHK) GO TO 50

IF(EXT(6) .LT. 60) GO TO 80

CALL TOPLIN
WRITE(6,75)
FORAT(IH)

ON(Z) = KN(28) * I
WRITE(6,90) (HT, NEXTI), EFIT(I), (I1,K,2)

CONTINUE

GO TO 50

IF (ZM) (.LT. 60) GO TO 80

CALL TOPLIN
WRITE(6,75)
FORAT(IH)
90 FORMAT(1X, A1, 16, (A4, B12.5, 1S1))
61.     A2 = A2 + 1
62.     J = J + 1
63.     RETURN
64.     E0 TO 60
65. 100 WRITE(6, 110) NDIM
66. 110 FORMAT(15E6) * = INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR SUBR
67.       WRITE(TPRINT, NOIM = 15, TH = * * 1
68.       STOP
69.       END.
APPENDIX B

USERS DESCRIPTION FOR PLOTA

This appendix presents user descriptions for a SINDA plotting routine, PLOTA, and a tape combining routine, MCOMB. Both routines are on the second file of the SINDA/Version 9 program tape but are main routines rather than user subroutines. A brief description of the routines and the user input description is given below.

PLOTA DESCRIPTION

The plot routine which is available on the SINDA program tape can be used with a history tape from a previous SINDA run to generate microfilm output. The items available for plotting are (1) pressures for each pressure node or pressure drop values for each tube, (2) valve positions for each valve, (3) flowrates for each tube and (4) temperatures for each temperature lump. Each of these items may be plotted as a function of mission time. The user specifies the grid time range to be plotted, a time label, and the items to be plotted. A number of history tapes may be combined prior to plotting the results. The user has the option of averaging any portion of the plotted curve and of specifying the range of the ordinate axis.

The system control cards and the data input cards for PLOTA are described below:

SYSTEM CONTROL CARDS FOR PLOTA

```
7 8 RUN
7 8 MSG
7 8 PLT
7 8 ASG A=XXX (SINDA PROGRAM TAPE)
7 8 ASG E=XXX (FIRST TAPE TO BE COMBINED)
7 8 ASG F=XXX (SECOND TAPE TO BE COMBINED)
```

Add additional ASG cards as required for tapes to be combined.
ASG T=XXX (COMBINED TAPE)
XQT CUR
TRW A
PEF A
IN A
TRI A
XQT PLOTA
DATA CARDS
EOF

PLOTA DATA CARDS

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Format</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Card 1 (Title Card)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-72</td>
<td>12A6</td>
<td>TITLEA</td>
<td>Any 72 alphabetic characters to be used as heading for each frame of plots</td>
</tr>
<tr>
<td>Card 2 (Parameter Card)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-10</td>
<td>F10.0</td>
<td>TA</td>
<td>First value of time to be plotted (hours).</td>
</tr>
<tr>
<td>11-20</td>
<td>F10.0</td>
<td>TZ</td>
<td>Last value of time to be plotted (hours).</td>
</tr>
<tr>
<td>21-30</td>
<td>F10.0</td>
<td>TPG</td>
<td>Time range for each grid. Number of grids drawn will be (TZ-TA)/TPG. (If TPG is left blank, the job will terminate.)</td>
</tr>
<tr>
<td>31-35</td>
<td>I5</td>
<td>ITMX</td>
<td>Time scale label:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 1, &quot;SECONDS&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 2, &quot;MINUTES&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 3, &quot;HOURS&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Any other value, &quot;*****&quot;</td>
</tr>
<tr>
<td>36-40</td>
<td>I5</td>
<td>MPNT</td>
<td>Print control code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 1, prints information to be plotted while loading the plot tape</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>≠ 1, will not print information to be plotted</td>
</tr>
</tbody>
</table>

B-2
<table>
<thead>
<tr>
<th>Columns</th>
<th>Format</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>41-45</td>
<td>I5</td>
<td>NTP</td>
<td>Number of tapes to be combined. Use a negative number if start and/or stop times are specified on Card 3 for any tape to be combined.</td>
</tr>
<tr>
<td>46-50</td>
<td>I5</td>
<td>KT</td>
<td>Logical unit number to which tape to be plotted is assigned. If left blank, unit 23 is assumed. (See Table B-I) The combined tape is assigned to this unit.</td>
</tr>
</tbody>
</table>
| 51-55   | I5     | INC   | \(=1\), every time point and associated data value from the tapes to be combined will be transferred to the combined tape. 
\(=2\), every second time point and associated data values will be transferred to the combined tape. 
etc. |
| 56-60   | I5     | IUNIT | Logical unit number to which first tape to be combined is assigned. If left blank, unit 7 is assumed. |
| 61-70   | F10.0  | ASTRT | Beginning time for averages (hours). |
| 71-80   | F10.0  | ASTOP | Ending time for averages (hours). |

**Card 3** (Required only if NTP < 0. See Card 2 columns 41-45)

| 1-5     | F5.3   | XSTART | First time point from first tape to be combined which will be transferred to the combined tape. |
| 6-10    | F5.3   | XSTOP  | Last time point from first tape to be combined which will be transferred to the combined tape. |

Repeat XSTART and XSTOP in five column fields for each tape to be combined.

**Card 4** (Item Card)

| 1-5     | I5     | ITEM   | The item number to be plotted. Use a negative value if this item is to start a |
A new grid. A maximum of four curves may be plotted on one grid. Insert a blank card when the number of items exceeds 34000 divided by the number of points between TA and TZ. More item cards may follow this blank card.

6-7 A2 ITYPE A two character item type code which determines the type of item to be plotted.
- XX, pressures or pressure drop values
- VP, valve positions
- FR, flow rates
- ST, node temperatures

8 I1 IREL = 0, node numbers are actual numbers
- 1, node numbers are relative numbers

9-10 I2 KAVG > 0, calculate the numerical average of this item over the interval specified by ASTRT and ASTOP in columns 61-70 and 71-80 of Card 2.
- 9, plot the average on the frame with this item.

11-58 8A6 TITLES Item description to be printed at the top of each grid, along with the plotting symbol which is generated and used by the program.

59-60 Blank

The next two values are optional (may be left blank) on the cards whose item numbers are negative and are ignored on all other item cards.

61-70 F10.0 YLØ The minimum (reference) value on the Y-axis

71-80 F10.0 YHI The maximum value on the Y-axis

The above limits will be replaced by the program if numbers outside this range are found in the histories of the items to be plotted on this grid.
Repeat Card 4 for each item to be plotted.

Card 5
1-80
Blank

Card 6
1-80
Blank

If additional history tapes are to be plotted, repeat Card 1 and subsequent cards for each additional history tape.

**COMBINE ROUTINE DESCRIPTION**

The combine routine, MCOMB, can be used to combine as many as six history tapes into one history tape prior to its being plotted or being compared to another tape. The combined tape which is generated can be saved for future use if required. The user selects the frequency with which the time points and associated data values on the original tapes are added to the new tape. That is, every time point on the original tape can be added to the new tape or every second, third, etc., point can be added depending on the requirements for the combined tape.

The combine routine is a very useful feature if several history tapes are generated on a long mission run. By combining these tapes before plotting, a continuous plot of the mission can be obtained. The convenience of the combine routine can also be observed when mission runs made with different time increments are compared. Obviously, the run made with the smaller time increment will take more computer time than the run made with the larger time increment, and will probably require at least one "restart". In such a situation, there would be two history tapes with the smaller time increment to compare to one with the larger time increment. Normally, this would take two separate runs. However, with the new arrangement, the two tapes with the smaller time increment can be combined and then compared to the tape with the larger time increment on the same run.

The system control cards and the data input cards for MCOMB are described below.
USER'S MANUAL FOR MCOMB ROUTINE

CONTROL AND PROGRAM CARDS

7Z RUN

7N MSG

7S ASG A=XXX (SINDA PROGRAM TAPE)

7 ASG E=XXX (First Tape to be combined)

7 ASG F=XXX (Second tape to be combined)

Add additional ASG cards as required for

tapes to be combined.

7 ASG K=XXX (Combined Tape)

7 XQT CUR

TRW A

PEF A

IN A

TRI A

7 XQT MCOMB

DATA CARDS

7 EOF
<table>
<thead>
<tr>
<th>Columns</th>
<th>Format</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Card 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-5</td>
<td>I5</td>
<td>NTAPE</td>
<td>Number of tapes to be combined. Use a negative value if start and/or stop times are specified on Card 2.</td>
</tr>
<tr>
<td>6-10</td>
<td>I5</td>
<td>IUNIT</td>
<td>Logical unit number to which first tape to be combined is assigned. If left blank unit 7 is assumed.</td>
</tr>
<tr>
<td>11-15</td>
<td>I5</td>
<td>KT</td>
<td>Logical unit number to which combined tape is assigned. If left blank, unit 13 is assumed.</td>
</tr>
<tr>
<td>16-20</td>
<td>I5</td>
<td>KODE2</td>
<td>= 1, time to be added to the times read from tapes to be combined will be supplied on Card 3.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 0, transfer times as read from tapes to be combined.</td>
</tr>
<tr>
<td>21-25</td>
<td>I5</td>
<td>INC</td>
<td>= 1; every time point and associated data values read from the tapes to be combined will be transferred to the combined tape.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 2, every second time point and associated data values will be transferred to the combined tape.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>etc.</td>
</tr>
<tr>
<td>Card 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-5</td>
<td>F5.3</td>
<td>XSTART</td>
<td>First time point from first tape to be combined which will be transferred to the combined tape.</td>
</tr>
<tr>
<td>6-10</td>
<td>F5.3</td>
<td>XSTOP</td>
<td>Last time point from first tape which will be transferred to the combined tape.</td>
</tr>
<tr>
<td>Columns</td>
<td>Format</td>
<td>Title</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Repeat XSTART and XSTOP in five columns fields for each tape to be combined.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Card 3 (Required only if KØDE2 &gt; 0. See Card 1 columns 16-20)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-10</td>
<td>F10.0</td>
<td>ADD</td>
<td>Time to be added to each time read from first tape to be combined.</td>
</tr>
</tbody>
</table>

Repeat ADD in 10 column fields for each tape to be combined.
<table>
<thead>
<tr>
<th>FORTRAN UNIT NO.</th>
<th>I/O DEVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
</tr>
<tr>
<td>7</td>
<td>E</td>
</tr>
<tr>
<td>8</td>
<td>F</td>
</tr>
<tr>
<td>9</td>
<td>G</td>
</tr>
<tr>
<td>10</td>
<td>H</td>
</tr>
<tr>
<td>11</td>
<td>I</td>
</tr>
<tr>
<td>12</td>
<td>J</td>
</tr>
<tr>
<td>13</td>
<td>K</td>
</tr>
<tr>
<td>14</td>
<td>L</td>
</tr>
<tr>
<td>15</td>
<td>M</td>
</tr>
<tr>
<td>16</td>
<td>N</td>
</tr>
<tr>
<td>18</td>
<td>O</td>
</tr>
<tr>
<td>19</td>
<td>P</td>
</tr>
<tr>
<td>20</td>
<td>Q</td>
</tr>
<tr>
<td>21</td>
<td>R</td>
</tr>
<tr>
<td>22</td>
<td>S</td>
</tr>
<tr>
<td>23</td>
<td>T</td>
</tr>
<tr>
<td>24</td>
<td>U</td>
</tr>
<tr>
<td>25</td>
<td>V</td>
</tr>
<tr>
<td>26</td>
<td>W</td>
</tr>
<tr>
<td>27</td>
<td>X</td>
</tr>
<tr>
<td>28</td>
<td>Y</td>
</tr>
<tr>
<td>29</td>
<td>Z</td>
</tr>
</tbody>
</table>