

AD-778 445

THE MARK IV SUPERSONIC-HYPERSONIC
ARBITRARY-BODY PROGRAM. VOLUME III.
PROGRAM LISTINGS

Arvel E. Gentry, et al

Douglas Aircraft Company

Prepared for:

Air Force Flight Dynamics Laboratory

November 1973

**Reproduced From
Best Available Copy**

DISTRIBUTED BY:

NTIS

**National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151**

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

ACCESSION FOR

PROJ	PROJ SECTION	<input checked="" type="checkbox"/>
...	SUB SECTION	<input type="checkbox"/>

...

A

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER AFFDL-TR-73-159 VOLUME III	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER AD 778 445	
4. TITLE (and Subtitle) THE MARK IV SUPERSONIC-HYPERSONIC ARBITRARY-BODY PROGRAM, VOLUME III PROGRAM LISTINGS		5. TYPE OF REPORT & PERIOD COVERED Final	
7. AUTHOR(s) Arvel E. Gentry Douglas N. Smyth Wayne R. Oliver		6. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Douglas Aircraft Company, McDonnell Douglas Corp. 3855 Lakewood Blvd. Long Beach, California 90846		8. CONTRACT OR GRANT NUMBER(s) F33615-72-1675	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Flight Dynamics Laboratory AFFDL/FXG Wright-Patterson AFB, Ohio		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE November 1973	
		13. NUMBER OF PAGES 578	
		15. SECURITY CLASS. (of this report) Unclassified	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
18. SUPPLEMENTARY NOTES <div style="text-align: center;"> <small> NATIONAL TECHNICAL INFORMATION SERVICE U.S. Department of Commerce Springfield, VA 22161 </small> </div>			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)			
Aerodynamics Arbitrary Three-Dimensional Shapes		Supersonic Hypersonic Flow Computers (Digital)	
		FORTRAN Graphics	
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)			
<p>This report describes a digital computer program system that is capable of calculating the supersonic and hypersonic aerodynamic characteristics of complex arbitrary three-dimensional shapes. This program is identified as the Mark IV Supersonic-Hypersonic Arbitrary-Body Computer Program. This program is a complete reorganization and expansion of the old Mark III Hypersonic Arbitrary-Body Program. The Mark IV program has a number of new capabilities that extend its applicability down into the supersonic speed range.</p>			

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 9102-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. ABSTRACT (continued)

The outstanding features of this program are its flexibility in covering a very wide variety of problems and the multitude of program options available. The program is a combination of techniques and capabilities necessary in performing a complete aerodynamic analysis of supersonic and hypersonic shapes. These include: vehicle geometry preparation; computer graphics to check out the geometry; analysis techniques for defining vehicle component flow field effects; surface streamline computations; the shielding of one part of a vehicle by another; calculation of surface pressures using a great variety of pressure calculation methods including embedded flow field effects; and computation of skin friction forces and wall temperature.

Although the program primarily uses local-slope pressure calculation methods that are most accurate at hypersonic speeds, its capabilities have been extended down into the supersonic speed range by the use of embedded flow field concepts. This permits the first order effects of component interference to be accounted for.

The program is written in FORTRAN for use on CDC or IBM types of computers.

The program is documented in three volumes. Volume I is primarily a User's Manual, Volume II gives the Program Formulation, and Volume III contains the Program Listings.

ia

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

AFFDL-TR-73-159 VOLUME III

**THE MARK IV SUPERSONIC-HYPERSONIC
ARBITRARY BODY PROGRAM**

VOLUME III – PROGRAM LISTINGS

*ARVEL E. GENTRY
DOUGLAS N. SMYTH
WAYNE R. OLIVER*

DOUGLAS AIRCRAFT COMPANY
McDONNELL DOUGLAS CORPORATION

Approved for public release; distribution unlimited.

ic

FOREWORD

This report describes a computer program developed at the Douglas Aircraft Division of the McDonnell Douglas Corporation, Long Beach, California. The development of the Douglas Arbitrary-Body Aerodynamic Computer Program was started in 1964 and greatly expanded in subsequent years under sponsorship of the Douglas Independent Research and Development Program (IRAD). From August 1966 to May 1967 the program development was continued under Air Force Contract No. F3361567-C-1008. The product of this work was the Mark II version of the program as released for use by government agencies in May 1967. Between 1967 and 1968 further Douglas IRAD work and another Air Force Contract (F33615-67-C1602) produced the Mark III Hypersonic Arbitrary-Body Program. The latest version of the program as presented in this report is identified as the Mark IV Supersonic-Hypersonic Arbitrary-Body Computer Program and was prepared in the period of 1972-73 under Air Force Contract F33615-72-C-1675. This contract was administered by the Air Force Flight Dynamics Laboratory, Flight Mechanics Division, High Speed Aero Performance Branch. The Air Force Project Engineers for this study were Verle V. Bland Jr., and Captain Hugh Wilbanks, AFFDL/FXG.

At the Douglas Aircraft Company, this work was conducted under the direction of Mr. Arvel E. Gentry as Principal Investigator. A number of major parts of the new program were prepared by Mr. Douglas N. Smyth. Mr. Wayne R. Oliver's work in applying the various versions of this program to practical design problems contributed both in program design and in program validation. A number of other people contributed to the various phases of this work for which the authors are grateful.

This report was submitted by the authors in November 1973.

This technical report has been reviewed and is approved.



PHILIP P. ANTONATOS
Chief, Flight Mechanics Division
Air Force Flight Dynamics Laboratory

ABSTRACT

This report describes a digital computer program system that is capable of calculating the supersonic and hypersonic aerodynamic characteristics of complex arbitrary three-dimensional shapes. This program is identified as the Mark IV Supersonic-Hypersonic Arbitrary-Body Computer Program. This program is a complete reorganization and expansion of the old Mark III Hypersonic Arbitrary-Body Program. The Mark IV program has a number of new capabilities that extend its applicability down into the supersonic speed range.

The outstanding features of this program are its flexibility in covering a very wide variety of problems and the multitude of program options available. The program is a combination of techniques and capabilities necessary in performing a complete aerodynamic analysis of supersonic and hypersonic shapes. These include: vehicle geometry preparation; computer graphics to check out the geometry; analysis techniques for defining vehicle component flow field effects; surface streamline computations; the shielding of one part of a vehicle by another; calculation of surface pressures using a great variety of pressure calculation methods including embedded flow field effects; and computation of skin friction forces and wall temperature.

Although the program primarily uses local-slope pressure calculation methods that are most accurate at hypersonic speeds, its capabilities have been extended down into the supersonic speed range by the use of embedded flow field concepts. This permits the first order effects of component interference to be accounted for.

The program is written in FORTRAN for use on CDC or IBM type computers.

The program is documented in three volumes. Volume I is primarily a User's Manual, Volume II gives the Program Formulation, and Volume III contains the Program Listings.

CONTENTS

	Page
SECTION I	SUMMARY 1
SECTION II	DATA STORAGE TECHNIQUES 2
	Unit 4 Geometry Data 3
	Unit 9 Force Data Storage 6
	Unit 10 Flow Field Data Storage 8
SECTION III	IBM GO-DECK SETUP 11
SECTION IV	OVERLAY STRUCTURE 12
	CDC Overlay 12
	IBM Overlay 14
APPENDIX A	PROGRAM LISTINGS A-1
	Index to Subroutine Listings A-2

Preceding page blank

SECTION I SUMMARY

This volume contains the source language listings of the Mark IV Supersonic-Hypersonic Arbitrary-Body Program (Mod 0 Version). The program as shown in this listing will operate on CDC 6500, 6600, and CYBER 74 computers. With a small converter program, the Mark IV program can be converted for operation on IBM 360 and 370 types of computers. This converter program is included with the listings.

The actual tapes released for this program may contain minor differences from these listings. Therefore, these listings should be used for reference and communication purposes only. Any problems, bugs, etc., encountered with this program should be brought to the attention of the authors as soon as possible. Also, because of the complexity and size of the program, users should consult with the authors before making extensive program additions or modifications. This will sometimes save time since the proposed changes may have already been accomplished by someone else.

SECTION II

DATA STORAGE TECHNIQUES

Each major part of the Mark IV program makes use of data that has been generated and stored by some other program component. The manner in which these data are stored and retrieved is important both to the user and to the programmer seeking to add some new capability to the program. The principal concern for the user is that he know what is stored, and what sequence number, pointer, etc., might be required as subsequent input data to retrieve it. The primary storage units are tabulated below.

<u>Unit Number</u>	<u>Name</u>	<u>Usage</u>
1	TAPEIN	Copy of all input data.
2	TAPEB	Scratch unit (used in SHIELD and GENCUT).
3	TAPEA	Negative elements from shielding analysis.
4	-	Geometry data storage (mass storage).
5	TAPEIN	System input data stream (copied to unit 1 in routine MONITR, TAPEIN set = 1).
6	TAPEOT	System output print unit.
7	TAPEG	System punch output unit.
8	-	Element data geometry data storage (Type 3 cards)
9	-	Force data storage (mass storage)
10	-	Flow field data storage (mass storage)
11	TAPEC	Scratch unit (used in AUXILI)
12	TAPEE	Geometry data for interference usage.
13	TAPEF	Geometry data plus local flow field data.
14	TAPED	Scratch.

Units 4, 9, and 10 listed above are identified as being "mass storage" units. That is, the units are used to store data along with the necessary tags or pointers so that a given item can be retrieved directly without using a sequential read. On the CDC machine this process is called "mass storage". On the IBM machines it is identified as "direct access" storage. Although the two machines use different type of FORTRAN statements to write and read the data, the general procedure is still the same. A pointer or tag must be set or calculated by the program for each FORTRAN write statement. This pointer or tag is the sequential number of the record in the data set. The use of the same tag number in a FORTRAN read will result in the proper retrieval of the record. The specific data stored on the mass storage units 4, 9, and 10 are described below.

Unit 4. Geometry Data

Unit 4 is the storage unit for geometry data generated in routine GEOM. The unit normally contains both the input element corner points plus all the characteristics of the plane quadrilateral. When it is requested to do so, the program will also store on unit 4 the surface property data calculated in the FORCE routine. Unit 4 has a master table of contents followed by the data for each element. The unit is set up and initialized in routine GEOM.

<u>Record Number</u>	<u>Word Number</u>	<u>Description</u>
1	1	Configuration title (CONFIG from the Geometry Control Card)
2	1	NEXT, pointer to next empty record.
	2	NP, number of PANELS currently stored.
	3	NPMAX, maximum number of panels provided for in the table of contents.
	4	NREM, number of record spaces provided for use on each element (set = 3 in GEOM).
3	1	Pointer to next record after geometry data storage is complete.
4		Not used.
5		Panel #1 table array.
	1	Panel number (ISTAT3 = 1).
	2	Panel name (PANEL from the Panel Identification Card)
	3	Record number of first element (ISTART).
	4	Number of elements in the panel (L).
	5	Panel orientation (IORN)
	6	Symmetry flag (SYMFCT)
	7	Number of columns of elements (N)
	8	Number of rows in each column (I)
	9	Panel number adjacent to side 1 of panel (NADJ1)
	10	Panel number adjacent to side 2 of panel (NADJ2)
	11	Panel number adjacent to side 3 of panel (NADJ3)
12	Panel number adjacent to side 4 of panel (NADJ4)	
6		Blank
7		Blank
8		Blank
9		Blank
10		Panel #2 table array (same format as record number 5)
11		Blank
12		Blank
13		Blank
14		Blank
15		Panel #3 table array (same format as record number 5)
etc.		

<u>Record Number</u>	<u>Word Number</u>	<u>Description</u>
(NP*5)		Panel #NP (last panel loaded)
(NP*5)+1		Blank
(NP*5)+2		Blank
(NP*5)+3		Blank
(NP*5)+4		Blank
(NPMAX*5)+1		Last panel for which space is provided
(NPMAX*5)+2		(NPMAX may be = NP if desired)
(NPMAX*5)+3		
(NPMAX*5)+4		
[(NPMAX+1)*5]		Element #1. First available record for element data. The array ELEM is written here in GEOM.
[(NPMAX+1)*5+1]		Pointer array (25 words) to where surface output data is written by FORCE for input side of vehicle. First word is for 1st α - β , 2nd for 2nd α - β , etc.
[(NPMAX+1)*5+2]		Same as above for reflected side of vehicle.

The data written on Unit 4 for each element are contained in the element data array, ELEM, in subroutine GEOM. As indicated above, the 25 word array ELEM for the first element is written on Unit 4 at record number [(NPMAX+1)*5]. In routine GEOM the next two records, [(NPMAX+1)*5+1] and [(NPMAX+1)*5+2], are filled with 0.0 values. These records will be filled with live pointer data in subroutine FORCE as indicated above. These pointers will give the location of the calculated force data. We thus have three records per element on Unit 4.

The element data array, ELEM, contains the following information:

ELEM(1)	=	L	Element number
ELEM(2)	=	N	Column number
ELEM(3)	=	I	Row number
ELEM(4)	=	NX	X-component of unit normal
ELEM(5)	=	NY	Y-component of unit normal
ELEM(6)	=	NZ	Z-component of unit normal
ELEM(7)	=	XCENT	X-coordinate of element centroid
ELEM(8)	=	YCEN	Y-coordinate of element centroid
ELEM(9)	=	ZCENT	Z-coordinate of element centroid
ELEM(10)	=	AREA	Element area
ELEM(11)	=	XIN(1)	X-coordinates of input element corner points
ELEM(12)	=	XIN(2)	
ELEM(13)	=	XIN(3)	
ELEM(14)	=	XIN(4)	
ELEM(15)	=	YIN(1)	Y-coordinates of input element corner points
ELEM(16)	=	YIN(2)	
ELEM(17)	=	YIN(3)	
ELEM(18)	=	YIN(4)	

ELEM(19)	=	ZIN(1)	Z-coordinates of input element
ELEM(20)	=	ZIN(2)	corner points
ELEM(21)	=	ZIN(3)	
ELEM(22)	=	ZIN(4)	
ELEM(23)	=	0.0	
ELEM(24)	=	0.0	
ELEM(25)	=	0.0	

When directed by the input parameter ISAVE subroutine FORCE will write certain results from the pressure and force calculations out on Unit 4. These surface property data are in the form of single records containing the array E1 in FORCE. Since up to 20 α - β conditions can be analyzed on a pass into FORCE, pointers are required on Unit 4 to indicate where each E1 record is written. These pointers are written in the vacant record locations provided right behind the original element data record, ELEM, for each element. The surface property data contained in the array E1 in FORCE are as follows.

E1(1)	=	IG4	Pointer to original element data on Unit 4
E1(2)	=	E(1)	Element number
E1(3)	=	0.0	
E1(4)	=	0.0	
E1(5)	=	CP	Element pressure coefficient
E1(6)	=	DELTA	Element impact angle,
E1(7)	=	SX	X-component of surface unit velocity vector
E1(8)	=	SY	Y-component of surface unit velocity vector
E1(9)	=	SZ	Z-component of surface unit velocity vector
E1(10)	=	BS(2)/PFS	Local pressure divided by freestream pressure
E1(11)	=	BS(3)/TFS	Local temperature divided by freestream temperature
E1(12)	=	DELCA	ΔC_A
E1(13)	=	DELCY	ΔC_Y
E1(14)	=	DELCLN	ΔC_N
E1(15)	=	DELCLL	ΔC_l
E1(16)	=	DELCLM	ΔC_m
E1(17)	=	DELCLN	ΔC_n
E1(18)	=	0.0	
E1(19)	=	0.0	
E1(20)	=	0.0	
E1(21)	=	0.0	
E1(22)	=	0.0	
E1(23)	=	0.0	
E1(24)	=	0.0	
E1(25)	=	0.0	

The total number of records that can be stored on Unit 4 is controlled by the pointer array size provided for in the MAIN program. In the initial release of the program this was set at 3000 records. A check is made in routine FORCE to prevent the storage of more than this number of

records. Thus, the total number of elements that can be analyzed at one time depends upon whether or not surface property data are saved.

The total number of record spaces required on Unit 4 is given by the following relationship.

$$L_{\text{total}} = 4 + \text{NPMAX} * 5 + L_{\text{max}} * 3 + L_{\text{save}}$$

where

NPMAX = Maximum number of panels provided for on the unit (input on the Geometry Control card).

L_{max} = The total number of elements for all panels.

L_{save} = The total number of elements for which local surface property data will be saved in routine FORCE. Must include the total element count for both sides of the vehicle if it is yawed and the symmetry reflection process is used.

The size of the pointer storage array for Unit 4 in the MAIN program should be increased from 3000 up to 6000 to 10000 if additional core space is available on the user's computer installation.

Unit 9. Force Data Storage

Unit 9 is used to store component force data generated in the inviscid routine PRES, and in the viscous routine VISCUS. Unit 9 contains a table of contents followed by the data for each vehicle component.

<u>Record Number</u>	<u>Word Number</u>	<u>Description</u>
1	1	NTOT, total number of components.
	2	ISUM, total number of summations.
	3	NEXT, next empty record space.
	4	TITLE9(1) = 10HFORCE DATA for CDC = 4HFORC for IBM
	5	TITLE9(2) = 10HSAVE UNIT for CDC = 4HE DA for IBM
	6	TITLE9(3) = blank for CDC = 4HTA S for IBM
	7	TITLE9(4) = blank for CDC = 4HAVE for IBM
	8	TITLE9(5) = blank for CDC = 4HUNIT for IBM

<u>Record Number</u>	<u>Word Number</u>	<u>Description</u>
2	1	Identification flag for component data (=0.0 for force data only, = 1.0 if derivatives are included).
	2-41	LCOM, pointers to location of component data (40 total).
3	1	Identification flag for summation data (=0.0 for force data only, = 1.0 if derivatives are included).
	2-41	LSUM, pointers to location of summation data (40 total).
4	1	Mach, freestream Mach number.
	2	V, freestream velocity.
	3	RENO, freestream Reynolds number per foot.
	4	ALT, altitude.
	5	PSTAG, stagnation pressure in atmospheres.
	6	TSTAG, stagnation temperature.
	7	GTYPE(1), gas type name.
	8	GTYPE(2), gas type name.
	9	SREF, reference wing area.
	10	SPAN, reference span length.
	11	MAC, reference mean aerodynamic chord.
	12	XCG, X-location of cg.
	13	YCG, Y-location of cg.
	14	ZCG, Z-location of cg.
5-9		Not used.
10		Force data table of contents for 1st component (record number is = LCOM(1) = 10).
	1 2-11	Component number = 1. IPANL array, panel numbers used to make up this component.
11	1	NAB, number of α - β sets.
	2	Value of α for 1st α - β set.
	3	Value of α for 2nd α - β set.
	etc.	
	21	Value of β for 1st α - β set.
	22	Value of β for 2nd α - β set.
	etc. 41	
12		Force data for 1st component at 1st α - β . The array F in routine PRES.
	1	F(1) = α
	2	F(2) = C_D
	3	F(3) = C_L
	4	F(4) = C_A
	5	F(5) = C_Y
	6	F(6) = C_N

<u>Record Number</u>	<u>Word Number</u>	<u>Description</u>
	7	F(7) = β
	8	F(8) = C_L/C_D
	9	F(9) = C_m
	10	F(10) = C_l
	11	F(11) = C_n
13	1-11	Force derivatives for 1st component at 1st α - β . Not used.
14	1-11	Force data for 1st component at 2nd α - β . Array F.
15	1-11	Force derivatives for 1st component at 2nd α - β . Not used.
16	1-11	Force data for 1st component at 3rd α - β . Array F.
17	1-11	Force derivatives for 1st component at 3rd α - β . Not used.
		etc.

The above storage process starting with record 10 is repeated for each vehicle component. The starting record number for each component is placed in the LCOM array that is stored in record number 2. The pointers to summation data sets are placed in LSUM and stored in record number 3. Normally, record number 4 containing the Mach number and other data is based on the last component placed on the unit.

Unit 10. Flow Field Data Storage

Unit 10 is the most complex of the mass storage (direct access) units used by the program. Although Unit 10 is called the Flow Field unit it is used to store more than just flow field data. It is also used to store surface property data, streamline data, and skin friction computation results. The unit contains a number of directories and tables that hold not only basic identification information, but pointers that provide storage location record numbers pointing to lower level directories and the various types of data.

The general format of Unit 10 is shown in Figure 1. Starting with the Master Director at the top we see that it contains a set of 5 pointers in the array IMTAB. These point to what are called major data sets. For example, each set could be for a completely different vehicle or for a different Mach number for the same vehicle. Each of these sets has a Set Directory containing the Mach number and a pointer array, LOAB, pointing to a different lower level director for each α - β combination. In turn, each α - β director has pointers, LORG, that point to the Region Directories. The Region Directories contain some basic flag and interpolation parameters plus a set of pointers used to reach and retrieve data in the still lower level sub-regions and secondary flow field tables.

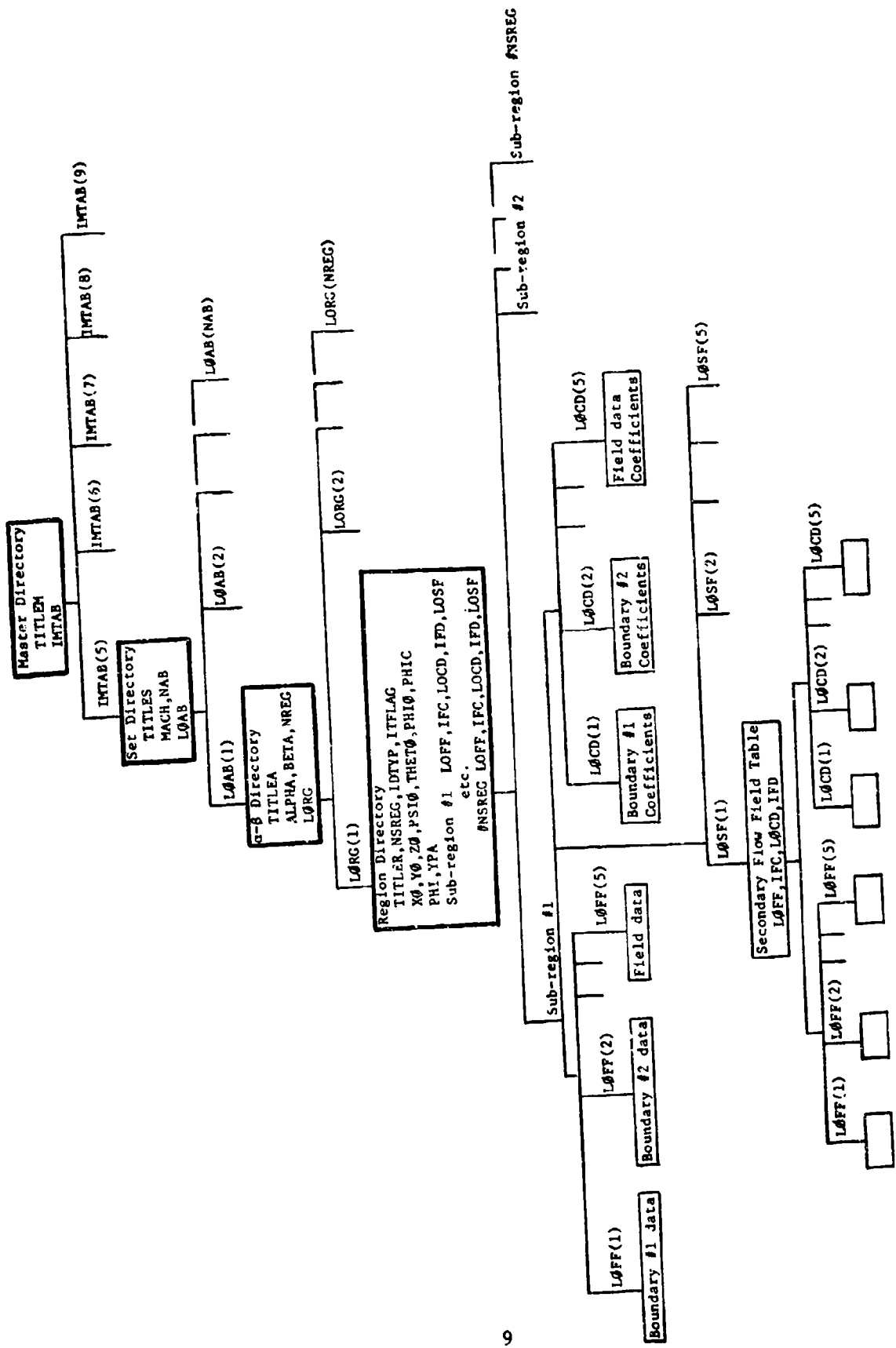


Figure 1. Flow Field Data Storage Format.

The basic format for each of the directory tables is listed below.

Record Number	Description
<u>Master Directory</u>	
1	TITLEM = Unit title (10 words)
2	IMTAB(1) = Number of separate data sets stored on the unit (NDSET) IMTAB(2) = Pointer to next empty record (LCNEXT) IMTAB(3) = Empty IMTAB(4) = Empty IMTAB(5) = Pointer to start of 1st set = LOSET(1) IMTAB(6) = Pointer to start of 2nd set = LOSET(2) IMTAB(7) = Pointer to start of 3rd set = LOSET(3) IMTAB(8) = Pointer to start of 4th set = LOSET(4) IMTAB(9) = Pointer to start of 5th set = LOSET(5)
3	Empty
<u>Set Directory</u> (First set is at record number 4)	
LOSET	TITLES, MACH, NAB (number of α - β sets)
LOSET+1	LOAB array pointers to α - β sets
LOSET+2	Empty
<u>α-β Directory</u>	
LOAB	TITLEA, ALPHA, BETA, NREG (number of regions)
LOAB+1	LORG array pointers to region directories
LOAB+2	Empty
<u>Region Directory</u>	
LORG	TITLER NSREG = Number of sub-regions IDTYP(1) = Data type flag = 1 flow field data = 2 surface property data = 3 streamline data = 4 shock shape data (not used at present time) IDTYP(2) = Uniform - non-uniform flag = 1 uniform flow field data = 2 non-uniform data ITFLAG Not used
LORG+1	XO, YO, ZO, PSIO, etc.
LORG+2	PHI or YPA (not active)
LORG+3	PHI or YPA (not active)
LORG+4	Empty
LORG+5	Pointers for sub-region #1 LOFF, IFC, LOCD, IFD, LOSF
LORG+6	Pointers for sub-region #2 LOFF, IFC, LOCD, IFD, LOSF
etc.	etc.
LORG+(4+NSRG)	Pointers for last sub-region (NSREG)

SECTION III
IBM GO-DECK SETUP

A suggested set of DD cards for operation of the Mark IV program on IBM 360 and 370 computers is given below.

```
// EXEC FORTGO, REGION=240K
//STEPLIB DD DSN=SYS3.C2BC, DISP=(SHR, PASS), UNIT=3330,
//          VOLUME=SER=CSLB07
//GO:FT01F001 DD DSN=&FT01, UNIT=SYSDA, SPACE=(TRK,(5,1)),
//          DCB=(RECFM=F, LRECL=80, BLKSIZE=80)
//GO.FT02F001 DD DSN=&FT02, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.FT03F001 DD DSN=&FT03, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.FT04F001 DD DSN=&FT04, UNIT=SYSDA, SPACE=(100,(3000)),
//          DCB=(BUFNO=1, DSORG=DA)
//GO.FT07F001 DD SYSOUT=B
//GO.FT08F001 DD DSN=&FT08, UNIT=SYSDA, SPACE=(TRK,(25,5)),
//          DCB=(RECFM=F, LRECL=80, BLKSIZE=80)
//GO.FT09F001 DD DSN=&FT09, UNIT=SYSDA, SPACE=(164,(500)),
//          DCB=(BUFNO=1, DSORG=DA)
//GO.FT10F001 DD DSN=&FT10, UNIT=SYSDA, SPACE=(100,(2000)),
//          DCB=(BUFNO=1, DSORG=DA)
//GO.FT11F001 DD DSN=&FT11, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.FT12F001 DD DSN=&FT12, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.FT13F001 DD DSN=&FT13, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.FT14F001 DD DSN=&FT14, UNIT=SYSDA, SPACE=(TRK,(10,5)),
//          DCB=(BUFNO=1, BLKSIZE=6447, RECFM=VBS)
//GO.SYSUDUMP DD SYSOUT=A, SPACE=(TRK,(100), RLSE)
//GO.SYSIN DD *
```

SECTION IV
OVERLAY STRUCTURE

Because of the large size of this program it is necessary to use the overlay features of FORTRAN for normal operation. The overlay structure for the CDC version of the program (CDC SCOPE 3.3 overlay) is shown below. With this structure the program requires approximately 114,100 octal words on the CDC machine (CYBER 74, FTN V3.0-P308 Opt=1).

CDC Overlay

OVERLAY(MARK4,0,0)

MAIN
HEADER
MONTR
ARSIN
ARCOS

OVERLAY(MARK4,1,0)

GEOM
PUNCH

OVERLAY(MARK4,1,1)

IELE

OVERLAY(MARK4,1,2)

ELLIP

OVERLAY(MARK4,1,3)

CUBIC

OVERLAY(MARK4,1,4)

AIRCFT
SURF
NACEL
FUSE
STATS3

OVERLAY(MARK4,2,0)

AERO
COMPR
EXPAND
CONE
NEWTPM
ATMOS
SOLVIT
GETT
SAVE
ROWFM1

OVERLAY(MARK4,2,1)

FLOW
MERID
NCONE
CONEA
MOUT
CADA
FFSPEC

FFAIR
FFSURF
FFINPT
FFBODY
SOSE
OVERLAY (MARK4,2,2)
SHIELD
OVERLAY (MARK4,2,3)
PRES
FORCE
SHKEXP
BLUNT
CPINPT
FLNTRP
VALU2
SFNTR2
ERF
ERRF
ACONE
OVERLAY (MARK4,2,4)
VISCUS
FORSF
SKINFR
TEMP
ROMU
QC
POLY
BLOCK DATA
CFINPT
INTEG
PRECAL
SFNTR3
VALU3
LAMNAR
PROFIL
TURBLN
RUNKUT
FUNCT
INT1
INT2
LGRNGE
ROOT
CURVFT
GRADNT
SMIHNA
SIMPS1
SPLINE
OVERLAY (MARK4,2,5)
SPEC
SUM
OVERLAY (MARK4,2,6)
STREAM
SFNTRP
VALUE

OVERLAY (MARK4, 3, 0)
GRAPH
OVERLAY (MARK4, 4, 0)
AUXILI
GENCUT
OUTD

IBM Overlay

The overlay for the IBM version of the program is shown below.
With this structure the program requires approximately 240K bytes
(FORTRAN IV H Opt-2, Level 20.1, OS/MVT Release 201).

INSERT MAIN, HEADER, EROR, MONITR
INSERT EXEC, TAPP, TAGS
OVERLAY ALPHA
INSERT GEOM
INSERT GFLAG
OVERLAY BETA
INSERT IELE, ELLIP, CUBIC, PUNCH
OVERLAY BETA
INSERT AIRCFT, SURF, FUSE, NACEL, STATS3
INSERT ACFT, ARFOIL, FUSD, NACELD
OVERLAY ALPHA
INSERT AERO, ATMOS
INSERT FLIGHT, GASD, ABDATA, REF, INTERF, FSBS
INSERT SHKEXP, COMPR, EXPAND, NEWTPM, CONE, BLUNT, GDATA, FARRAY, MDATA
INSERT SOLVIT, ROWFMI, SAVE, GETT
OVERLAY BETA
INSERT FLOW, FFINPT, FFBODY, FFAIRF, FFSPEC, FFSURF
INSERT SOSE, NCONE, CONEA, MERID, MOUT
INSERT FFIELD, SOSEIN, SOSEOT, CADA
OVERLAY BETA
INSERT SHIELD
OVERLAY BETA
INSERT PRES, FORCE, ACONE
INSERT FLNTRP, CPINPT, SFNTR2, SURFN2, VALU2
OVERLAY BETA
INSERT VISCUS, FORSF, TEMP, QC, POLY, PROP, ROMU, TEMPQC, SKIND, C1
OVERLAY BAMA
INSERT SKINFR
OVERLAY GAMA
INSERT INTEG, RUNKUT, GRADNT, LGRNGE, FUNCT, INT1, INT2, SMTHNA, SPLINE
INSERT ROOT, CURVFT, SIMPS1
OVERLAY DELTA
INSERT PRECAL
OVERLAY DELTA
INSERT LAMNAR
OVERLAY DELTA
INSERT TURBLN, PROFIL

OVERLAY GAMA
 INSERT CFINPT, SFNTR3, VALU3
OVERLAY BETA
 INSERT SPEC, SUM
OVERLAY BETA
 INSERT STREAM
 INSERT SFNTRP, VALUE, SURFNT
OVERLAY ALPHA
 INSERT GRAPH
OVERLAY ALPHA
 INSERT AUXILI
 INSERT GENCUT, OUTD

APPENDIX A
PROGRAM LISTINGS

This part of the report contains the source card listings for the Mark IV Mod 0 Version of the Supersonic-Hypersonic Arbitrary-Body Program. The listing as presented here is for the CDC version of the program. This program has been run on CDC 6500, 6600, and CYBER 74 models. The program is written in FORTRAN for the CDC FTN compiler (opt=1) and has been run under the SCOPE 3.3 and 3.4 operating systems.

In this listing all cards that are peculiar to the CDC version of FORTRAN are identified by a C in card column 80. All cards that are peculiar to the IBM FORTRAN IV compiler are identified by an I in card column 80 and a C in card column 1. In other words, the code for both CDC and IBM machines is in the deck but the IBM cards are made inactive by converting them to comment statements (C in card column 1). Since all of the machine dependent cards are identified by an I or C in card column 80 it is a simple matter to convert the deck from one version to the other with a small conversion program. When converting from CDC to IBM code this conversion program reads and copies each card to a storage unit. If a card has a C in card column 80, then a C is written into card column 1 to make the CDC peculiar card inactive. If a card has an I in card column 80, then the C is removed from card column 1 and the card image written to the storage unit as an active FORTRAN statement. The conversion from IBM back to CDC is made in a similar manner.

The conversion program to convert the deck from CDC to IBM FORTRAN is listed below (for use on an IBM machine).

```
DIMENSION DATA(22)
DATA CB,CC,CI/1H ,1HC,1HI/
REWIND 19
DO 100 I=1,20000
  READ (5,20,END=300) DATA
20  FORMAT (1A1,19A4,1A2,1A1)
  IF (DATA(22) .EQ. CI) DATA(1) = CB
  IF (DATA(22) .EQ. CC) DATA(1) = CC
  WRITE (19,20) DATA
100 CONTINUE
300 STOP
END
```

This program places the new deck with IBM cards made active, and CDC cards inactive, on to unit 19. The references to unit 19 above can be changed to unit 7 to punch the deck out.

INDEX TO SUBROUTINE LISTINGS

<u>Routine</u>	<u>Name</u>	<u>Deck</u>	<u>Page</u>
MAIN	Main Program (Executive Program)	MAIN	5
MONITR	Input Data Read and Print Routine	MONI	10
HEADER	Page header subroutine	HEAD	12
EROR	IBM ABEND dump routine	EROR	13
ARSIN	Arcsine conversion routine (CDC)	ARSI	14
ARCOS	Arccosine conversion routine (CDC)	ARCO	15
GEOM	Geometry Data Preparation and Control	GEOM	16
PUNCH	Element data write subroutine	PUNC	29
IELE	Element data transfer routine	IELE	31
ELLIP	Elliptical cross-section routine	ELLI	33
CUBIC	Parametric cubic routine	CUBI	41
AIRCFE	Aircraft geometry executive program	AIRC	49
SURF	General airfoil surface routine	SURF	62
NACEL	Nacelle geometry generation routine	NACE	73
FUSE	Fuselage geometry generation routine	FUSE	77
STATS3	Dummy zero area element routine	STAT	82
AERO	Aerodynamic analysis executive routine	AERO	83
COMPR	Compression routine	COMP	88
EXPAND	Prandtl-Meyer expansion routine	EXPA	92
CONE	Conical flow routine	CONE	97
NEWTPM	Blunt body Newtonian + Prandtl-Meyer	NEWT	102
ATMOS	Atmosphere subprogram	ATMO	108
SOLVIT	Matrix operation routine	SOLV	112
GETT	Data retrieval routine for SOLVIT	GETT	125
SAVE	Data save routine for SOLVIT	SAVE	126
ROWFM1	Matrix formation for surface spline	ROWF	127
FLOW	Executive routine for flow field analysis	FLOW	130
MERID	Meridian cutting plane routine	MERI	140
NCONE	Improved cone flow routine	NCON	155
CONEA	Cone at angle of attack routine	CONE	158
MOUT	Meridian plane output data organization	MOUT	163
CADA	Circular arc data analysis routine	CADA	172
FFSPEC	Simple flow field data input routine	FFSP	175
FFAIRF	Dummy routine for future use	FFAI	179
FFSURF	Surface data transfer routine	FFSU	180
FFINPT	Flow field data hand load routine	FFIN	188
FFBODY	Body flow field data generation routine	FFBO	196
SOSE	Second-order shock-expansion routine	SOSE	206
SHIELD	Shielding routine	SHIE	223
PRES	Executive routine for pressure calculations	PRES	250
FORCE	Inviscid pressure and force calculation	FORC	261
SHKEXP	Element strip shock expansion routine	SHKE	284
BLUNT	Blunt-body viscous routine	BLUN	289
CPINPT	CP surface interpolation routine	CPIN	293
FLNTRP	Flow field interpolation routine	FLNT	304
VALU2	Spline interpolation routine	VAL2	326
SFNTR2	Surface data interpolation routine	SFN2	332

<u>Routine</u>	<u>Name</u>	<u>Deck</u>	<u>Page</u>
ERF	Error function routine (CDC)	ERF	342
ERRF	Error function calculation (CDC)	ERRF	343
ACONE	Cone at angle of attack	ACON	344
VISCUS	Viscous executive routine	VISC	350
FORSF	Skin friction force routine	FORS	360
SKINFR	Mark III skin friction calculation routine	SKIN	370
TEMP	Surface temperature calculation routine	TEMP	381
ROMU	Density-viscosity routine	ROMU	389
QC	Heating at given wall temperature routine	QC	394
POLY	N-th order polynomial routine	POLY	398
BLOCK	Block data routine for skin friction	BLKD	399
CFINPT	Skin friction data interpolation control	CFIN	402
INTEG	Executive routine for integral boundary lay.	INTE	414
PRECAL	Preliminary calculation routine	PREC	429
SFNTR3	Skin friction interpolation routine	SFN3	435
VALU3	Skin friction interpolation	VAL3	445
LAMNAR	Integral method laminar boundary layer	LAMN	452
PROFIL	Velocity profile calculation	PROF	464
TURBLN	Turbulent boundary layer routine	TURB	469
RUNKUT	Integration of turbulent equations	RUNK	472
FUNCT	Integral boundary layer functions	FUNC	477
INT1	Integral method function 1	INT1	479
INT2	Integral method function 2	INT2	480
LGRNGE	Lagrangian interpolation	LGRN	482
ROOT	Root finding routine	ROOT	484
CURVFT	Curve fit routine	CURV	485
GRADNT	Curve gradient routine	GRAD	486
SMTHNA	Curve smoothing routine	SMTH	487
SIMPS1	Simpsons rule integration routine	SIMP	488
SPLINE	Spline curve fit routine	SPLI	490
SPEC	Special function routine	SPEC	491
SUM	Force data summation routine	SUM	492
STREAM	Streamline calculation routine	STRE	500
SFNTRP	Surface data interpolation	SFNT	516
VALUE	Spline interpolation routine	VALU	526
GRAPH	Graphics executive routine	CRAP	532
AUXILI	Auxiliary methods executive routine	AUXI	533
GENCUT	General cutting plane routine	GENC	535
OUTD	Cutting plane output data collection	OUTD	551

MAIN

```

OVERLAY(MARK4,0,0)
PROGRAM MAIN (INPUT=129,OUTPUT,PUNCH=129,TAPES=INPUT,TAPE=OUTPUT,
1 TAPE7=PUNCH,TAPE1=129,TAPE2=129,TAPE3=129,TAPE4=129,TAPE=129,
2 TAPE=129,TAPE10=129,TAPE11=129,TAPE12=129,TAPE13=129,
3 TAPE14=129)
*****
C** SUPERSONIC-HYPECSONIC ARBITRARY-RUDY AERODYNAMIC COMPUTER PROGRAM **
C** CDC/IBM 360 MODEL.
*****
C THIS IS THE MARK IV MOD 0 VERSION OF THE SUPERSONIC-HYPERSONIC
C ARBITRARY-BODY FORCE ANALYSIS PROGRAM, THE BASIC PROGRAM
C WAS DEVELOPED AT THE DOUGLAS AIRCRAFT COMPANY, AIRCRAFT
C DIVISION UNDER SPONSORSHIP OF THE AIR FORCE FLIGHT DYNAMICS LAB.
C
C***** THIS PROGRAM WAS WRITTEN BY A. E. GENTRY AND D. N. SMYTH *****
C*****
C
C***** EXECUTIVE MAIN PROGRAM *****
C
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /TAPE/TAPEIN,TAPEOUT,TAPFA,TAPB,TAPC,TAPD,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON INDEX4(3000),INDEX9( 500),INDEX10(2000)
DIMENSION IPG(20),TITLE(15)
INTEGER TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
INTEGER PAGE,ERROR
EXTERNAL EROR
DEFINE FILE 4 (3000,25,U,ITAG4)
DEFINE FILE 9 (500,41,U,ITAG9)
DEFINE FILE 10(2000,25,U,ITAG10)

```

MAIN 001C
 MAIN 002C
 MAIN 003C
 MAIN 004C
 MAIN 005C
 MAIN 006
 MAIN 007
 MAIN 008
 MAIN 009
 MAIN 010
 MAIN 011
 MAIN 012
 MAIN 013
 MAIN 014
 MAIN 015
 MAIN 016
 MAIN 017
 MAIN 018
 MAIN 019
 MAIN 020
 MAIN 021
 MAIN 022
 MAIN 023
 MAIN 024
 MAIN 025
 MAIN 026
 MAIN 027C
 MAIN 028
 MAIN 029
 MAIN 030
 MAIN 031
 MAIN 032I
 MAIN 033I
 MAIN 034I
 MAIN 035I

MAIN

Preceding page blank

MAIN

MAIN 036C
 MAIN 037C
 MAIN 038C
 MAIN 039
 MAIN 040
 MAIN 041
 MAIN 042
 MAIN 043
 MAIN 044
 MAIN 045
 MAIN 046
 MAIN 047
 MAIN 048
 MAIN 049
 MAIN 050
 MAIN 051
 MAIN 052
 MAIN 053
 MAIN 054
 MAIN 055
 MAIN 056
 MAIN 057
 MAIN 058
 MAIN 059
 MAIN 060
 MAIN 061Y
 MAIN 062Y
 MAIN 063
 MAIN 064
 MAIN 065
 MAIN 066
 MAIN 067
 MAIN 068
 MAIN 069
 MAIN 070

CALL OPENMS (4,INDEX4,3001,0)
 CALL OPENMS (9,INDEX9,501,0)
 CALL OPENMS (10,INDEX10,2001,0)
 ERROR = 0

PAGE = 1
 TAPFIN = 5
 TAPEID = 6
 TAPEA = 3
 TAPEB = 2
 TAPEC = 11
 TAPEE = 14
 TAPEF = 12
 TAPEG = 13
 TAPEH = 7
 REWIND TAPEA
 REWIND TAPEB
 REWIND TAPEC
 REWIND TAPEE
 REWIND TAPEF
 REWIND TAPEG

```

C *****
C ***** READ EXECUTIVE FLAG CARD *****
C ***** READ (TAPEIN,10) IEROR,INMONT
      10 FORMAT (2I1)
C IF (IEROR .EQ. 0) CALL ERRSET (207,256,256,2,EROR,301)
C CALL ERRSET(208,256,0,0,1)
C IF (INMONT .EQ. 0) CALL MONITR
C *****
C ***** READ SYSTEM CONTROL CARD TO SELECT PROGRAMS TO BE USED AND ORDER
C ***** OF USE. A MAXIMUM OF 20 OPTIONS MAY BE USED. THIS VERSION OF
C ***** THE PROGRAM WILL ALLOW ONLY OPTIONS 1, 2, 3, 4 *****
C *****
C READ (TAPEIN,20) IPG,CASE
  
```

MAIN

MAIN

```

20 FORMAT (20I1,4I1,A2)
WRITE (6,30)
30 FORMAT (1H1,////,1H ,39H***SUPERSONIC-HYPERSONIC ARBITRARY-BODY
1 50H AERODYNAMIC COMPUTER PROGRAM SYS'FM ***** ,//,1H ,
2 10X,46HPROGRAM OPTIONS ARE IN THE FOLLOWING ORDER.... ,//1H )
IF (IPG(1) .EQ. 0) GO TO 220

DO 80 I=1,20
IF (IPG(I) .EQ. 0) GO TO 90
IF (IPG(I) .GT. 4) GO TO 240
IF (IPG(I) .EQ. 1) WRITE (TAPEOT,40) I
40 FORMAT (1H0,15X,12,30H GEOMETRY PROGRAM (OPTION 1) )
IF (IPG(I) .EQ. 2) WRITE (TAPEOT,50) I
50 FORMAT (1H0,15X,12,33H AFDYDYNAMIC PROGRAM (OPTION 2) )
IF (IPG(I) .EQ. 3) WRITE (TAPEOT,60) I
60 FORMAT (1H0,15X,12,30H GRAPHICS PROGRAM (OPTION 3) )
IF (IPG(I) .EQ. 4) WRITE (TAPEOT,70) I
70 FORMAT (1H0,15X,12,31H AUXILIARY PROGRAMS (OPTION 4) )
80 CONTINUE
90 I = 0

C 100 I = I + 1
IF (IPG(I) .EQ. 0) GO TO 260

C IPRG = IPG(I)
GO TO (110,120,130,140),IPRG

C 110 CALL GEOM
110 CALL OVERLAY (5HMARK4,1,0,6HRECALL)
GO TO 150

C 120 CALL AFRO
120 CALL OVERLAY (5HMARK4,2,0,6HRECALL)
GO TO 150

C

```

```

MAIN 071
MAIN 072
MAIN 073
MAIN 074
MAIN 075
MAIN 076
MAIN 077
MAIN 078
MAIN 079
MAIN 080
MAIN 081
MAIN 082
MAIN 083
MAIN 084
MAIN 085
MAIN 086
MAIN 087
MAIN 088
MAIN 089
MAIN 090
MAIN 091
MAIN 092
MAIN 093
MAIN 094
MAIN 095
MAIN 096
MAIN 097
MAIN 098I
MAIN 099C
MAIN 100
MAIN 101
MAIN 102I
MAIN 103C
MAIN 104
MAIN 105

```

MAIN

MAIN

```

C 130 CALL GRAPH
130 CALL OVERLAY (SHMARK4,3,0,6HRECALL)
GO TO 150
C
C 140 CALL AUXILI
140 CALL OVERLAY (SHMARK4,4,0,6HRECALL)
GO TO 150
C
150 IF (ERROR .NE. 0) GO TO 200
WRITE (TAPEOT,160)
160 FORMAT (1H1,///,1H0,40H***** MAIN PROGRAM NOW HAS CONTROL OF
1 16H SYSTEM ***** )
IF (ERROR .EQ. 0) GO TO 100
IF (ERROR .EQ. 2) GO TO 200
C
C WRITE (TAPEOT,170) I, ERROR
170 FORMAT (1M0,36H*****AN ERROR HAS OCCURRED IN PHASE ,I3,
1 61H AND WAS DETECTED AFTER RETURN TO THE MAIN EXECUTIVE ROUTINE.,
2 9H ERROR #,I2)
C
C 200 WRITE (TAPEOT,210)
210 FORMAT (1H ,//,1H ,40H***** FATAL ERROR *** PROG. M STOPPED )
GO TO 280
C
C
C 220 WRITE (TAPEOT,230)
230 FORMAT (1M ,//,1M ,38H***** FIRST PHASE OPTION IS ZERO *****
1 21H***** FATAL ERROR **** )
GO TO 280
C
C 240 WRITE (TAPEOT,250)
250 FORMAT (1M ,//,1M ,42H***** PROGRAM OPTION IS GREATER THAN 4 ***
1 22H***** FATAL ERROR **** )

```

```

MAIN 106I
MAIN 107C
MAIN 108
MAIN 109
MAIN 110I
MAIN 111C
MAIN 112
MAIN 113
MAIN 114
MAIN 115
MAIN 116
MAIN 117
MAIN 118
MAIN 119
MAIN 120
MAIN 121
MAIN 122
MAIN 123
MAIN 124
MAIN 125
MAIN 126
MAIN 127
MAIN 128
MAIN 129
MAIN 130
MAIN 131
MAIN 132
MAIN 133
MAIN 134
MAIN 135
MAIN 136
MAIN 137
MAIN 138
MAIN 139
MAIN 140

```

MAIN

MAIN

MAIN 141
MAIN 142
MAIN 143
MAIN 144
MAIN 145
MAIN 146
MAIN 147

```
C      GO TO 260
      260 WRITE (TAPENT,270)
      270 FORMAT (1H ,//,1H ,44H***** PROGRAM HAS REACHED NORMAL TERMINATION
      1 7H ***** )
      280 STOP
      END
```

MAIN

MONITR

MONI 001
 MONI 002
 MONI 003
 MONI 004
 MONI 005
 MONI 006
 MONI 007
 MONI 008
 MONI 009
 MONI 010
 MONI 011
 MONI 012
 MONI 013
 MONI 014
 MONI 015
 MONI 016
 MONI 017
 MONI 018
 MONI 019
 MONI 020
 MONI 021
 MONI 022
 MONI 023
 MONI 024
 MONI 025
 MONI 026
 MONI 027
 MONI 028
 MONI 029
 MONI 030
 MONI 031
 MONI 032
 MONI 033
 MONI 034
 MONI 035

```

SUBROUTINE MONITR
C THIS ROUTINE IS USED TO TRANSFER THE INPUT DATA TO A SCRATCH UNIT
C
C
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPFF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPFC,TAPED,TAPEE,TAPFF,
1 TAPEG,TAPEH,TAPEI,TAPFJ,TAPEK
INTEGER ERROR,PAGE,CASE
DIMENSION CARD(20),TITLF(15)
DATA BLANK/4H /
CASE = 0
DO 5 I=1,15
5 TITLF(I) = BLANK

C
10 CALL HEADER
WRITE (TAPEOT,20)
20 FORMAT (1H0,50H$UPERSONIC-HYPERSONIC ARBITRARY=BODY PROGRAM INPUT,
1 5H DATA ,/1H )
WRITE (TAPEOT,25)
25 FORMAT (1H ,49HC
1 31H5
2 60H12345678901234567890123456789012345678901234567890/)
I = 10

C
30 READ (TAPEIN,40,END=60) CARD
30 READ (TAPEIN,40) CARD
IF (EOF(5)) 40,41
40 FORMAT (20A4)
41 WRITE (1,40) CARD
50 WRITE (TAPEOT,50) CARD
50 FORMAT (1H ,20A4)
I = I + 1

```

MONITR

MONITR

MONI	036
MONI	037
MONI	038
MONI	039
MONI	040
MONI	041
MONI	042

```

IF (I .EQ. 51) GO TO 10
GO TO 30
C
60 REWIND 1
TAPEIN = 1
RETURN
END

```

MONITR

HEADER

```
C SURROUTINE HEADER
C
C COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
C INTEGER PAGE, ERROR, CASE
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C DIMENSION TITLE(15)
C
C PRINT OUT HEADER AT TOP OF EACH PAGE OF OUTPUT
WRITE (TAPEOT,10) CASE,PAGE,TITLE
;0 FORMAT (1H1,46HSUPERSONIC/HYPERSONIC ARBITRARY-BODY PROGRAM, /
1 13HMARK IV MOD 0,/,1H0,6HCASE ,2X,12,85X,5HPAGE ,14,/,1H0,1444,
2 1A3)
C
C STEP PAGE NUMBER BY ONE
PAGE = PAGE + 1
C
C RETURN
END
```

HEAD 001
HEAD 002
HEAD 003
HEAD 004
HEAD 005
HEAD 006
HEAD 007
HEAD 008
HEAD 009
HEAD 010
HEAD 011
HEAD 012
HEAD 013
HEAD 014
HEAD 015
HEAD 016
HEAD 017
HEAD 018
HEAD 019
HEAD 020
HEAD 021
HEAD 022
HEAD 023

HEADER

EROR

EROR	0011
EROR	002
EROR	003
EROR	004
EROR	005
EROR	006
EROR	007
EROR	0081
EROR	0091
EROR	010
EROR	0111
EROR	0121
EROR	0131
EROR	0141
EROR	0151
EROR	0161

```

C   SURROUTINE EROR
C
C *****
C THIS ROUTINE IS USED TO CAUSE AN ABEND DUMP FOR SYSTEM ERRORS
C AS DIRECTED BY THE FIRST DATA CARD.
C *****
C   DIMENSION A(2)
C
C   SET A PARAMETER THAT WILL USUALLY CAUSE AN ABEND
C     I = 100000
C     A(I) = 0.0
C     IF (I .GT. 1) STOP
C   RETURN
C   END

```

EROR

AR SIN

AR SI 001C
AR SI 002
AR SI 003
AR SI 004C
AR SI 005C
AR SI 006C

FUNCTION AR SIN(X)
C THIS ROUTINE IS REQUIRED BECAUSE OF DIFFERENCES BETWEEN CDC AND IBM
C FORTRAN.
AR SIN = ASIN(X)
RETURN
END

AR SIN

ARCOS

ARCO 001C
ARCO 002
ARCO 003
ARCO 004C
ARCO 005C
ARCO 006C

FUNCTION ARCOS(X)
C THIS ROUTINE IS REQUIRED BECAUSE OF DIFFERENCES BETWEEN CDC AND IBM
C FORTRAN.
ARCOS = ACOS(X)
RETURN
END

ARCOS

GEOM

```

OVERLAY (MARK4,1,0)
PROGRAM GEOM
SURROUTINE GEOM
C
C *** THIS ROUTINE CONTROLS THE EXECUTION OF THE GEOMETRY GENERATION
C *** ROUTINES
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /GFLAG/IOUT, ISTAT3, IORIN, COMPIH
COMMON /TAPE/TAPFIN, TAPEIN, TAPEA, TAPFB, TAPEC, TAPEE, TAPEE, TAPFF,
1 TAPFG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /TAGS/ITAG4, ITAG9, ITAG10
C
DIMENSION XA(250), XR(250), YA(250), YB(250), ZA(250), ZR(250),
1 XI(4), ETA(4), YIN(4), YIN(4), ZIN(4), TITL(15), XPA(4), YPA(4), ZPA(4)
DIMENSION CONFIG(15), IGEOM(10), E(25), EN(25), EP(25)
DIMENSION F FM(25), PANEL(50), IORN(50), SYMFCT(50), IFA(50)
C
REAL NX, NY, NZ, LEFT
LOGICAL RFLAG, AFLAG, RFLAG
INTEGER TAPFIN, TAPEIN, TAPEA, TAPFB, TAPEC, TAPEE, TAPEF,
1 TAPFG, TAPEH, TAPEI, TAPEJ, TAPFK
INTEGER STAT, STATI, PAGE, CASE, TYPE, ERROR, PRINTS, SYMFCT, SEQ
C
ISTAT3 = 0
C
C READ GEOMETRY CONTROL CARD
READ (TAPEIN,10) IOUT, IREM, PRINTS, IOUAD, ISMAX, NEW, NPMAX, CONFIG
10 FORMAT (I12,5I1,1I2,15A4)
C INITIALIZE ELEMENT DATA STORAGE UNIT IF REQUIRED
DO 13 I=1,25
E(I) = 0.0
FP(I) = 0.0
13 EO(I) = 0.0

```

GEOM

GEOM

GEOM 036
GEOM 037
GEOM 038I
GEOM 039C
GEOM 040
GEOM 041
GEOM 042
GEOM 043
GEOM 044
GEOM 045
GEOM 046
GEOM 047
GEOM 048
GEOM 049
GEOM 050
GEOM 051I
GEOM 052C
GEOM 053
GEOM 054I
GEOM 055C
GEOM 056
GEOM 057
GEOM 058
GEOM 059
GEOM 060
GEOM 061
GEOM 062
GEOM 063
GEOM 064
GEOM 065
GEOM 066
GEOM 067
GEOM 068
GEOM 069
GEOM 070

GEOM

```
IF (NEW .EQ. 1) GO TO 15
IGA = 1
WRITE (4,IG4) CONFIG
CALL WRITMS (4,CONFIG,15,IG4)
IF (NPMAX .EQ. 0) NPMAX = 50
C NREM IS THE NUMBER OF RECORDS PER ELEMENT
NREM = 3
NEXT = (NPMAX+1)*5
NP = 0
E(1) = NEXT
E(2) = NP
E(3) = NPMAX
E(4) = NREM
GO TO 19

15 IGA = 1
C READ (4,IG4) CONFIG
CALL READMS (4,CONFIG,15,IG4)
IGA = 2
READ (4,IG4) E
CALL READMS (4,E,25,IG4)
NEXT = E(1)
NF = E(2)
NPMAX = E(3)
NREM = E(4)
19 CONTINUE

C CHECK IF QUADRILATERALS ONLY ARE TO BE CALCULATED
IF (IGUAD .EQ. 1) GO TO 110
IF (IREM .EQ. 0 .AND. IOUT.NF.5 .AND. IOUT.NF.1) REMIND IOUT

C READ COMPONENT IDENTIFICATION CARD
20 ISTAT3 = ISTAT3 + 1
READ (TAPEIN,30) PAMEL(ISTAT3),LAST,ICRN4(ISTAT3),IGEOM,
1 SYMFC(ISTAT3),IFA(ISTAT3),NADJ1,NADJ2,NADJ3,NADJ4
30 FORMAT (A4,P11,10I1,2I1,4I2)
IURIEN = ICRN(ISTAT3)
```

GEOM

GEOM 071
GEOM 072
GEOM 073
GEOM 074
GEOM 075
GEOM 076
GEOM 077
GEOM 078
GEOM 079
GEOM 080I
GEOM 081C
GEOM 082
GEOM 083
GEOM 084I
GEOM 085C
GEOM 086
GEOM 087I
GEOM 088C
GEOM 089
GEOM 090I
GEOM 091C
GEOM 092
GEOM 093
GEOM 094
GEOM 095
GEOM 096
GEOM 097
GEOM 098
GEOM 099
GEOM 100
GEOM 101
GEOM 102
GEOM 103
GEOM 104
GEOM 105

```
      COMPIN = PANFL(ISTAT3)
C
C *****
C READ OR GENERATE ELEMENTS FOR EACH VEHICLE COMPONENT
      DO 90 I=1,10
      IF (IGEOM(I) .EQ. 0) GO TO 100
      IG = IGEOM(I)
      GO TO (40,50,60,70), IG
C 40 CALL IELE
      40 CALL OVERLAY (SHMARK4,1,1,6HRECALL)
      GO TO 80
C
C 50 CALL ELLIP
      50 CALL OVERLAY (SHMARK4,1,2,6HRECALL)
      GO TO 80
C 60 CALL CUBIC
      60 CALL OVERLAY (SHMARK4,1,3,6HRECALL)
      GO TO 80
C 70 CALL AIRCFT
      70 CALL OVERLAY (SHMARK4,1,4,6HRECALL)
C
C 80 IF (ERROR .NE. 0) GO TO 420
      90 CONTINUE
C
C 100 IF (LAST .EQ. 0) GO TO 20
      C THE GENERATION OF ALL ELEMENT DATA IS COMPLETED
      I3MAX = ISTAT3
C
C 110 IF (TQUAD .EQ. 2) GO TO 420
      IF (YOUT .NE. 5 .AND. INUT .NE. 1) REWIND IOJY
      IG4 = I3MAX + 1
C
C *****
```

GEOM

GEOM

```

C ELEMENT DATA WILL NOW BE CONVERTED TO QUADRILATERALS
C SET UP STARTING CONSTANTS
  ISTAT3 = 0
120 CONTINUE
  ISTAT3 = ISTAT3 + 1
  IF (ISTAT3 .LE. NPMAX) GO TO 125
  WRITE (TAPEOT,122)
122 FORMAT (1H0,46H**NUMBER OF PANELS EXCEEDS NPMAX**PROGRAM STOP )
  STOP
125 IFACT = IFA(ISTAT3)
  IORIEN = IORN(ISTAT3)
C
  N = -1
  NN = -1
  KLCY = 0
  NPRT = 10
  L = 0
  IGT = 0
  AREAT = 0.0
  VOL = 0.0
C READ SCALE FACTOR CARD IF REQUIRED
  IF (IFACT .EQ. 1) READ (TAPEIN,130) XSC,YSC,ZSC,DELX,DELY,DELZ
130 FORMAT (6F10.0)
  JA = 0
  JB = 0
  IF (IORIEN .EQ. 2) IB = 1
  IF (IORIEN .EQ. 3) IA = 1
  IF (IORIEN .GT. 1) IORIEN = 1
  IF (IGTYPE .GT. 0) IGT = 1
C
  READ IN ALL SURFACE DATA
  READ (ICUT,140) X,Y,Z,STAT,XX,YY,ZZ,STAT1,CASE,SECT,TYPE,SEQ
140 FORMAT(3F10.4,I1,3F10.4,I1,2X,I2,1A4,I2,4X,I4)
  STAT = IABS(STAT)
  STATT = IABS(STATT)

```

GEOM 106
GEOM 107
GEOM 108
GEOM 109
GEOM 110
GEOM 111
GEOM 112
GEOM 113
GEOM 114
GEOM 115
GEOM 116
GEOM 117
GEOM 118
GEOM 119
GEOM 120
GEOM 121
GEOM 122
GEOM 123
GEOM 124
GEOM 125
GEOM 126
GEOM 127
GEOM 128
GEOM 129
GEOM 130
GEOM 131
GEOM 132
GEOM 133
GEOM 134
GEOM 135
GEOM 136
GEOM 137
GEOM 138
GEOM 139
GEOM 140

GEOM

GEOM

```
IF (TYPE,NE,3) GO TO 430
IF ((STAT,EQ,0.OR,STAT,EG,0).AND,(STAT,NE,2.AND,STAT,NE,2))
1  SECTS = SECT
   RFLAG = .FALSE.
   GO TO 180
150 IF (RFLAG) GO TO 160
   RFLAG = .TRUE.
   X = XX
   Y = YY
   Z = ZZ
   STAT = STATT
   GO TO 170
160 RFLAG = .FALSE.
   READ (IOUT,140) X,Y,Z,STAT,XX,YY,ZZ,STATT,CASE,SECT,TYPE,SEQ
   STAT = IABS(STAT)
   STATT = IABS(STATT)
   IF (TYPE,NE,3) GO TO 430
   IF ((STAT,EQ,0.OR,STAT,EG,0).AND,(STAT,NE,2.AND,STAT,NE,2))
1  SECTS = SECT
   MC = M
170 IF (STAT,EG,0.UR,STAT,EQ,5) GO TO 260
   IF (STAT,EG,2) GO TO 270
   IF (.NOT. AFLAG) GO TO 270
   MC = M
180 M = 1
   IF (STAT,EQ,2) GO TO 240
   IF (.NOT. BFLAG) GO TO 220
190 DO 200 J=1,MC
   XA(J) = XB(J)
   YA(J) = YB(J)
200 7A(J) = ZB(J)
210 XB(1) = X
   YB(1) = Y
   ZB(1) = Z
   GO TO 150
220 IF (AFLAG) GO TO 230
```

GEOM 141
GEOM 142
GEOM 143
GEOM 144
GEOM 145
GEOM 146
GEOM 147
GEOM 148
GEOM 149
GEOM 150
GEOM 151
GEOM 152
GEOM 153
GEOM 154
GEOM 155
GEOM 156
GEOM 157
GEOM 158
GEOM 159
GEOM 160
GEOM 161
GEOM 162
GEOM 163
GEOM 164
GEOM 165
GEOM 166
GEOM 167
GEOM 168
GEOM 169
GEOM 170
GEOM 171
GEOM 172
GEOM 173
GEOM 174
GEOM 175

GEOM

GEOM

GEOM 176
GEOM 177
GEOM 178
GEOM 179
GEOM 180
GEOM 181
GEOM 182
GEOM 183
GEOM 184
GEOM 185
GEOM 186
GEOM 187
GEOM 188
GEOM 189
GEOM 190
GEOM 191
GEOM 192
GEOM 193
GEOM 194
GEOM 195
GEOM 196
GEOM 197
GEOM 198
GEOM 199
GEOM 200
GEOM 201
GEOM 202
GEOM 203
GEOM 204
GEOM 205
GEOM 206
GEOM 207
GEOM 208
GEOM 209
GEOM 210

GEOM

```
      BFLAG = .TRUE.  
      GO TO 190  
230  AFLAG = .FALSE.  
      GO TO 210  
240  AFLAG = .TRUF.  
      BFLAG = .FALSE.  
      N = N + 1  
      NN = NN + 1  
250  YA(M) = X  
      YA(M) = Y  
      ZA(M) = Z  
      GO TO 150  
260  M = M + 1  
      IF (AFLAG) GO TO 250  
      XR(M) = X  
      YR(M) = Y  
      ZR(M) = Z  
      IF (STAT .NE. 3) GO TO 150  
270  MMTN = MIN0 (M,MC) - 1  
      NN2 = 1  
      MC = M  
      N = N + 1  
      NN = NN + 1  
      KLCT = KLCT + 1  
      JJ = 0  
  
C     BEGIN COMPUTATION OF SURFACE FLEMPT CHARACTERISTICS  
      GO 390 I= 1,MMIN  
      IIA = I + IA  
      IIR = I + IB  
      IF (IFACT.EQ.1) GO TO 280  
          XIN(1) = XA(IIA )  
          XIN(2) = XA(IIA +1)  
          XIN(3) = XB(JIB +1)  
          XIN(4) = XB(JIR )
```

GEOM

GEOM 211
 GEOM 212
 GEOM 213
 GEOM 214
 GEOM 215
 GEOM 216
 GEOM 217
 GEOM 218
 GEOM 219
 GEOM 220
 GEOM 221
 GEOM 222
 GEOM 223
 GEOM 224
 GEOM 225
 GEOM 226
 GEOM 227
 GEOM 228
 GEOM 229
 GEOM 230
 GEOM 231
 GEOM 232
 GEOM 233
 GEOM 234
 GEOM 235
 GEOM 236
 GEOM 237
 GEOM 238
 GEOM 239
 GEOM 240
 GEOM 241
 GEOM 242
 GEOM 243
 GEOM 244
 GEOM 245

GEOM

```

YIN(1) = YA(IIA )
YIN(2) = YA(IIA +1)
YIN(3) = YP(IIB +1)
YIN(4) = YP(IIB )
ZIN(1) = ZA(IIA )
ZIN(2) = ZA(IIA +1)
ZIN(3) = ZR(IIR +1)
ZIN(4) = ZB(IIB )
GO TO 290

C
280 XIN(1) = XA(IIA ) * XSC + DELX
XIN(2) = XA(IIA +1) * XSC + DELX
XIN(3) = XB(IIB +1) * XSC + DELX
XIN(4) = XR(IIR ) * XSC + DELX
YIN(1) = YA(IIA ) * YSC + DELY
YIN(2) = YA(IIA +1) * YSC + DELY
YIN(3) = YB(IIB +1) * YSC + DELY
YIN(4) = YP(IIB ) * YSC + DELY
ZIN(1) = ZA(IIA ) * ZSC + DELZ
ZIN(2) = ZA(IIA +1) * ZSC + DELZ
ZIN(3) = ZB(IIR +1) * ZSC + DELZ
ZIN(4) = ZP(IIB ) * ZSC + DELZ

C FORM DIAGONAL VECTORS
290 T1X = XIN(3) * XIN(1)
T2X = XIN(4) * XIN(2)
T1Y = YIN(3) * YIN(1)
T2Y = YIN(4) * YIN(2)
T1Z = ZIN(3) * ZIN(1)
T2Z = ZIN(4) * ZIN(2)

C FORM CROSS PRODUCT NET2 X T1
NX = T2Y*T1Z - T1Y*T2Z
NY = T1X*T2Z - T2X*T1Z
NZ = T2X*T1Y - T1X*T2Y
VN = SQRT ( NX*NX + NY*NY + NZ*NZ )
  
```

GEOM

GEOM 246
GEOM 247
GEOM 248
GEOM 249
GEOM 250
GEOM 251
GEOM 252
GEOM 253
GEOM 254
GEOM 255
GEOM 256
GEOM 257
GEOM 258
GEOM 259
GEOM 260
GEOM 261
GEOM 262
GEOM 263
GEOM 264
GEOM 265
GEOM 266
GEOM 267
GEOM 268
GEOM 269
GEOM 270
GEOM 271
GEOM 272
GEOM 273
GEOM 274
GEOM 275
GEOM 276
GEOM 277
GEOM 278
GEOM 279
GEOM 280

GEOM

```

C      IF (VN .EQ. 0.0) GO TO 300
C      FORM UNIT NORMAL VECTOR
      NX = NX / VN
      NY = NY / VN
      NZ = NZ / VN
C      COMPUTE AVERAGE POINT
      300 AVX = 0.25 * (XIN(1) + XIN(2) + XIN(3) + XIN(4) )
      AVY = 0.25 * (YIN(1) + YIN(2) + YIN(3) + YIN(4) )
      AVZ = 0.25 * (ZIN(1) + ZIN(2) + ZIN(3) + ZIN(4) )
C      COMPUTE PROJECTION DISTANCE
      D = NX*(AVX - XIN(1)) + NY*(AVY - YIN(1)) + NZ*(AVZ-ZIN(1))
      PD = ABS(D)
C
      T = SQRT (TIX*TIX + TIY*TIY + TIZ*TIZ)
      IF (T .EQ. 0.0) GO TO 310
      TIX = TIX / T
      TIY = TIY / T
      TIZ = TIZ / T
C      310 T2X = NY*TIZ - NZ*TIY
      T2Y = NZ*TIX - NX*TIZ
      T2Z = NX*TIY - NY*TIX
C      COMPUTE COORDINATES OF CORNER POINTS IN REFERENCE COORD. SYSTEM
      DO 320 J = 1,4
      XPA(J) = XIN(J) + NX*D
      YPA(J) = YIN(J) + NY*D
      ZPA(J) = ZIN(J) + NZ*D
      D = - D
      XDIF = XPA(J) - AVX
      YDIF = YPA(J) - AVY
      ZDIF = ZPA(J) - AVZ

```

GEOM

```

C   TRANSFORM CORNER POINTS TO ELEMENT COORDINATE SYSTEM (XI,ETA) WITH
C   AVERAGE POINT AS ORIGIN
      XI(J) = T1X*XDIF + T1Y*YDIF + T1Z*ZDIF
320  ETA(J) = T2X*XDIF + T2Y*YDIF + T2Z*ZDIF
      FTACK = ETA(2) - ETA(4)
      IF (ETACK .NE. 0.0) GO TO 330
      XI0 = 0.0
      GO TO 340
C   COMPUTE CENTROID
330  XI0 = .33333333 * (XI(4) * (ETA(1)-ETA(2)) + XI(2)
      1      * (ETA(4)-ETA(1))) / (ETA(2)-ETA(4))
340  ETA0 = -.33333333 * ETA(1)
C   OBTAIN CORNER POINTS IN SYSTEM WITH CENTROID AS ORIGIN
      DO 350 J = 1,4
      XI(J) = XI(J) - XI0
350  FTA(J) = ETA(J) - ETA0
C   TRANSFORM CENTROID TO REFERENCE COORDINATE SYSTEM
      XCENT = AVX + T1X*XI0 + T2X*ETA0
      YCENT = AVY + T1Y*XI0 + T2Y*ETA0
      ZCFNT = AVZ + T1Z*XI0 + T2Z*ETA0
C   CONSTANTS FOR USE IN COMPUTING AREA OF ELEMENT
      XI3M1 = XI(3) + XI(1)
      ETA2M4 = ETA(2) - ETA(4)
C   COMPUTE AREA AND VOLUME OF ELEMENTS
      AREA = 0.5 * XI3M1 * ETA2M4
      AREAT = AREAT + AREA
      DELVOL = ARFA * NY * YCENT
      VOL = VOL + DELVOL
      L = L + 1

```

GEOM 281
 GEOM 282
 GEOM 283
 GEOM 284
 GEOM 285
 GEOM 286
 GEOM 287
 GEOM 288
 GEOM 289
 GEOM 290
 GEOM 291
 GEOM 292
 GEOM 293
 GEOM 294
 GEOM 295
 GEOM 296
 GEOM 297
 GEOM 298
 GEOM 299
 GEOM 300
 GEOM 301
 GEOM 302
 GEOM 303
 GEOM 304
 GEOM 305
 GEOM 306
 GEOM 307
 GEOM 308
 GEOM 309
 GEOM 310
 GEOM 311
 GEOM 312
 GEOM 313
 GEOM 314
 GEOM 315

GEOM

GEOM

GEOM 316
GEOM 317
GEOM 318
GEOM 319
GEOM 320
GEOM 321
GEOM 322
GEOM 323
GEOM 324
GEOM 325
GEOM 326
GEOM 327
GEOM 328
GEOM 329
GEOM 330
GEOM 331
GEOM 332
GEOM 333
GEOM 334
GEOM 335
GFOM 336
GEOM 337
GEOM 338
GEOM 339
GEOM 340
GEOM 341
GEOM 342
GEOM 343
GEOM 344
GEOM 345
GEOM 346
GEOM 347
GEOM 348
GEOM 349
GEOM 350

```
      II = I
      IF (PRINTS.EQ.0) GO TO 380
C
C
C   PRINT RESULTS OF CALCULATIONS TO DETERMINE ELEMENT CHARACTERISTICS
      IF (NPRT .GE.9) GO TO 360
         NPRT = NPRT + 1
      IF (I .EQ. 1) GO TO 370
      WRITE (TAPEOT,450) I,XIN,NX,XCFNT,AREA,L,YIN,NY,YCENT,DELVOL,
      1  ZIN,NZ,ZCENT,VOL
         GO TO 380
      360 NPRT = 0
         CALL HEADER
         WRITE (TAPEOT,480)
      370 WRITE (TAPEOT,460) N,I,XIN,NX,XCENT,AREA,L,YIN,NY,YCENT,DELVOL,
      1  ZIN,NZ,ZCENT,VOL
C
C   SET UP DATA TO BE SAVED AND USED IN FORCE CALCULATIONS
      380 FLEM(1) = L
          ELEM(2) = N
          ELEM(3) = I
          ELEM(4) = NX
          FLEM(5) = NY
          ELEM(6) = NZ
          FLEM(7) = XCENT
          ELEM(8) = YCFNT
          FLEM(9) = ZCENT
          ELEM(10) = AREA
          FLEM(11) = XIN(1)
          FLEM(12) = XIN(2)
          ELEM(13) = XIN(3)
          FLEM(14) = XIN(4)
          ELEM(15) = YIN(1)
          FLEM(16) = YIN(2)
          ELEM(17) = YIN(3)
```

GEOM

GEOM

GEOM 351
 GEOM 352
 GEOM 353
 GEOM 354
 GEOM 355
 GEOM 356
 GEOM 357
 GEOM 358
 GEOM 359
 GEOM 360
 GEOM 361
 GEOM 362
 GEOM 363
 GEOM 364
 GEOM 365
 GEOM 366
 GEOM 367
 GEOM 368
 GEOM 369
 GEOM 370
 GEOM 371
 GEOM 372
 GEOM 373
 GEOM 374
 GEOM 375
 GEOM 376
 GEOM 377
 GEOM 378
 GEOM 379
 GEOM 380
 GEOM 381
 GEOM 382
 GEOM 383
 GEOM 384
 GEOM 385

GEOM

```

ELFM(16) = YIN(4)
ELFM(19) = ZIN(1)
ELFM(20) = ZIN(2)
ELFM(21) = ZIN(3)
ELFM(22) = ZIN(4)
ELFM(23) = 0.0
ELFM(24) = 0.0
ELFM(25) = 0.0
C SAVE ELEMENT DATA ON GEOMETRY SAVE UNIT
IG4 = 1
C WRITE (IG4) FLEM
CALL WKTIME (4,ELEM,25,IG4)
IF (L.EQ.1) ISTART = IG4
IF (NREM.EQ.1) GO TO 385
NREM = NREM + 1
DO 384 J=1,NREM
IG4 = IG4 + 1
384 WRITE (4,IG4) F0
385 CALL WKTIME (4,E0,25,IG4)
385 NEXT = NEXT + NREM
390 CONTINUE
C
C CHECK TO IDENTIFY END OF INPUT ELEMENTS
400 IF (STAT.LT.2) GO TO 410
NPR = NPR + 1
WRITE (TAPEOT,970) SECTS,AREAT,4,VOL
NN = NN + 1
N = N + 1
C TEST FOR END OF CASE
410 IF (STAT.NE.3) GO TO 180
C
C RECORD TABLE OF CONTENTS DATA ON GEOMETRY SAVE UNIT
EP(1) = ISTAT3
EP(2) = PANEL(JSTAT3)
EP(3) = ISTART
  
```

GEOM

GEOM 386
 GEOM 387
 GEOM 388
 GEOM 389
 GEOM 390
 GEOM 391
 GEOM 392
 GEOM 393
 GEOM 394
 GEOM 395
 GEOM 396I
 GEOM 397C
 GEOM 398
 GEOM 399I
 GEOM 400C
 GEOM 401
 GEOM 402I
 GEOM 403
 GEOM 404
 GEOM 405I
 GEOM 406C
 GEOM 407
 GEOM 408I
 GEOM 409C
 GEOM 410
 GEOM 411
 GEOM 412
 GEOM 413
 GEOM 414
 GEOM 415I
 GEOM 416C
 GEOM 417
 GEOM 418
 GEOM 419I
 GEOM 420C

GEOM

EP(4) = L
 EP(5) = IORN(ISTAT3)
 EP(6) = SYMFC7(ISTAT3)
 EP(7) = N
 EP(8) = I
 EP(9) = NADJ1
 EP(10) = NADJ2
 EP(11) = NADJ3
 EP(12) = NADJ4
 IG4 = ISTAT3*5
 WRITE (4#IG4) EP
 CALL WRITMS (4,EP,25,IG4)
 IG4 = IG4 + 1
 WRITE (4#IG4) EO
 CALL WRITMS (4,EO,25,IG4)
 IG4 = IG4 + 1
 WRITE (4#IG4) FO
 CALL WRITMS (4,EO,25,IG4)
 IG4 = IG4 + 1
 WRITE (4#IG4) EO
 CALL WRITMS (4,EO,25,IG4)
 IG4 = IG4 + 1
 WRITE (4#IG4) FO
 CALL WRITMS (4,EO,25,IG4)
 IF (ISTAT3 .GT. NP) NP = ISTAT3
 IF (ISTAT3 .LT. I3MAX) GO TO 120
 E(1) = NEXT
 F(2) = NP
 IG4 = 2
 WRITE (4#IG4) E
 CALL WRITMS (4,E,25,IG4)
 FO(1) = NEXT
 IG4 = 3
 WRITE (4#IG4) EO
 CALL WRITMS (4,EO,25,IG4)

GEOM

```

C 020 GO TO 490
C 021 GEOM 421
C 022 GEOM 422
C 023 GEOM 423
C 024 GEOM 424
C 025 GEOM 425
C 026 GEOM 426
C 027 GEOM 427
C 028 GEOM 428
C 029 GEOM 429
C 030 GEOM 430
C 031 GEOM 431
C 032 GEOM 432
C 033 GEOM 433
C 034 GEOM 434
C 035 GEOM 435
C 036 GEOM 436
C 037 GEOM 437
C 038 GEOM 438
C 039 GEOM 439
C 040 GEOM 440
C 041 GEOM 441
C 042 GEOM 442
C 043 GEOM 443
C 044 GEOM 444
C 045 GEOM 445
C 046 GEOM 446
C 047 GEOM 447
C 048 GEOM 448
C 049 GEOM 449

040 FORMAT (1M0,5CM**** SURFACE DATA ROUTINE HAS ATTEMPTED TO READ A
142M NON SURFACE CARD * CHECK YOUR CARDS **** )
ERROR = 1
GO TO 490

C 0450 FORMAT (1M0,7X, I4, 1P4E14.5,0PF10.6,1P2E14.5,16.2(/12X,4E14.5,
1 0PF10.6,1P2E14.5) )

C 0460 FORMAT (1M0,3X, 2I4,1P4E14.5,0PF10.6,1P2E14.5,16.2(/12X,4E14.5,
1 0PF10.6,1P2E14.5) )

C 0470 FORMAT (1M0,10M SECTION =1A2.33M TOTAL AREA OF INPUT ELEMENTS =
1 F12.3,6X26HTOTAL NUMBER OF FLEMFNTS = I5/IH ,12X,
2 33M TOTAL VOLUME OF INPUT ELEMENTS =F12.3, /1M0,3C20X,
3 9M*****)

C 0480 FORMAT (1M0,28M INPUT SURFACE ELEMENT DATA/1M0,6X1M3X1HM7X1HX,
1 3(13X,1MX),11X2HNX9X5HXCENT9X4MAREAX1ML ,/1H ,5X, 4(13X,1MY),
2 11X2HNX9X5HXCENT ,/1X,7HDELTA V,/1M ,5X,4(13X,1MZ),11X2HNZ,
3 9X,5HZCENT ,7X,6HVOLUME,/1H )

C 0490 RETURN
C 0490 CONTINUE
END
```

GEOM

PUNCH

```

C SURROUTINE PUNCH (X1,Y1,Z1,NSTAT1,X2,Y2,Z2,NSTAT3,SECT,TYPE,
: LINE,SEC,LAST,IP,INT,NREC)
C THIS SUBROUTINE PREPARES VEHICLE GEOMETRY DATA IN THE PROPER FORM
C FOR USE BY SDATA ROUTINE
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /GFLAG/ICUT, ISTAT3, IORIN, COMPIN
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEJ, TAPEK, TAPEL,
1 TAPEM, TAPEH, TAPEI, TAPEJ, TAPEK
1 DIMENSION TITLE(15), SECT(1)
C
INTEGER PAGE, SEC, ERROR, CASE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEJ, TAPEL, TAPEM, TAPEH, TAPEI, TAPEK, TAPEJ, TAPEL
1 TAPEM, TAPEH, TAPEI, TAPEJ, TAPEK
C
NSTAT2 = NSTAT3
C
CHECK IF THIS IS THE LAST POINT OF THE ENTIRE VEHICLE
IF (NSTAT3.EQ.3 .AND. LAST.EQ.1) NSTAT2 = 0
IF (IPRINT .EQ. 0) GO TO 40
C
IF (LINE .LT. 50 ) GO TO 20
C
WRITE PAGE HEADER FOR STANDARD OUTPUT TAPE
WRITE (TAPEOT,10) CASE, (TITLE(L),L=1,12), PAGE
10 FORMAT (1H1,5X,36MANALYTICALLY GENERATED ELEMENT DATA ,/
1 1H0,6H CASE,3X,12,17X,12H4,17X,5HPAGE ,14,/
2 1H0,5X1HY9X1HZ4X1H5SX1HX9X1HY9X1HZ5X1H5X1H75X1H5X1H10H CASE SECT,
3 6X3HSEQ )
C
PAGE = PAGE + 1
LINE = 5
C
WRITE GEOMETRY CARDS ON STANDARD OUTPUT TAPE

```

PUNCH

PUNCH

PUNC 036
PUNC 037
PUNC 038
PUNC 039
PUNC 040
PUNC 041
PUNC 042
PUNC 043
PUNC 044
PUNC 045
PUNC 046
PUNC 047
PUNC 048
PUNC 049
PUNC 050
PUNC 051

```
20 WRITE (TAPEOY,10) X1,Y1,Z1,NSTAT1,X2,Y2,Z2,NSTAT2,CASE,SECT,  
1 TYPE,SEG  
30 FORMAT (I40,3F10.4,I1,3F10.4,I1,2X12,A4,1X11,4MAERO,I4 )  
C LINE = LINE + 2  
C  
C WRITE GEOMETRY DATA ON GEOMETRY TAPE  
40 WRITE (IOUT,50) X1,Y1,Z1,NSTAT1,X2,Y2,Z2,NSTAT2,CASE,SECT,  
1 TYPE,SFG  
50 FORMAT(3F10.4,I1,3F10.4,I1,2X12,A4,1X11,4MAERO,I4 )  
C NRFC = NRFC + 1  
C SEQ = SEQ + 1  
C RETURN  
C END
```

PUNCH

IELE

```

OVERLAY (MARKQ,1,1)
PROGRAM IELE
SUBROUTINE IFLE

C THIS ROUTINE IS USED TO TRANSFER ELEMENT DATA (TYPE 3 CARDS) FROM
C AN INPUT UNIT TO THE ELEMENT DATA SAVE UNIT

COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEO,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /EXEC/CASE,TITLE,PAGE,EPROR
COMMON /GFLAG/IOUT,ISTAT3,IORIN,COMPIN
DIMENSION TITLE(15),XYZ1(3),XYZ2(3)
INTEGER STAT,STATT,TYPE,CASE,ERROR,PAGE,SEQ
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEO,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK

C IS3 = 0
C READ ELEMENT CONTROL CARD
10 READ (TAPEIN,10) ISMAX,IN,IREW,IS
10 FORMAT (2I2,2I1)
IF (IN.EQ. 0) IN = TAPEIN
IF (IN.NE.5 .AND. IREW.EQ.1) REWIND IN

C WRITE (TAPEOT,20) IN,IOUT,ISMAX
20 FORMAT (I4,4HTYPE 3 CARDS ARE BEING TRANSFERRED FROM UNIT ,I2,
1 9H TO UNIT ,I2,5X,6HI3MAX=,I4)

C IF (ISMAX .EQ. 0) ISMAX = 1
C 30 READ (IN,40) XYZ1,STAT,XYZ2,STATT,CASE,SECT,TYPE,SEQ
40 FORMAT (3F10.4,I1,3F10.4,I1,2X,I2,1A0,I2,4X,I4)
IF (STAT .LE. 0) STAT = 0
IF (STATT .LE. 0) STATT = 0

```

IELE 001C
IELE 002C
IELE 003I
IELE 004
IELE 005
IELE 006
IELE 007
IELE 008
IELE 009
IELE 010
IELE 011
IELE 012
IELE 013
IELE 014
IELE 015
IELE 016
IELE 017
IELE 018
IELE 019
IELE 020
IELE 021
IELE 022
IELE 023
IELE 024
IELE 025
IELE 026
IELE 027
IELE 028
IELE 029
IELE 030
IELE 031
IELE 032
IELE 033
IELE 034
IELE 035

IELE

IELE

IELE	036
IELE	037
IELE	038
IELE	039
IELE	040
IELE	041
IELE	042
IELE	043
IELE	044
IELE	045
IELE	046
IELE	047
IELE	048
IELE	049
IELE	050
IELE	051

```

IF (STAT,NE,3 ,AND,STATT,NE,3) GO TO 50
IS3 = IS3 + 1
IF (IS3,EG,I3MAX ,AND, I3,EG,0) GO TO 40
IF (STAT,EG,3) STAT = 0
IF (STATT,EG,3) STATT = 0

C 50 WRITE (IOUT,40) XYZ1,STAT,XYZ2,STATT,CASE,SECT,TYPE,SEO
   IF (IS3 ,EG, I3MAX) GO TO 70
   GO TO 30

C 60 WRITE (IOUT,60) XYZ1,STAT,XYZ2,STATT,CASE,SECT,TYPE,SEO
C
C 70 RETURN
C 70 CONTINUE
END

```

IELE

ELLIP

```

      PLAY (MARK4,1,2)
PROGRAM ELLIP
SUBROUTINE ELLIP
THIS SUBROUTINE PREPARES THE REQUIRED SURFACE ELEMENTS FOR
CIRCULAR OR ELLIPTICAL ARCS. EACH CROSS-SECTION
IS CONSIDERED SEPARATELY. DUMMY POINTS ARE COMPUTED SO THAT EACH
SECTION IS FORCED TO HAVE AN EVEN NUMBER OF POINTS AND SO THAT
POINTS IN A ROW ARE CORRECTLY MATCHED WITH POINTS IN AN ADJACENT
ROW WHEN THESE ROWS CONTAIN AN UNEQUAL NUMBER OF POINTS,

C THE PARAMETER DISCON WHICH IS SPECIFIED BY THE PROGRAMMER IS VALUED
  DEPENDING ON HOW THE POINTS ARE TO BE MATCHED
  DISCON= 1 ALL THETA AND THETA ARE THE SAME. DELTETH MUST
  DIVIDE THE ANGULAR INCREMENT THETA - THETA0
  EVENLY.
  = 2 ALL THETA ARE EQUAL BUT THETA VARIES
  = 3 ALL THETA ARE EQUAL BUT THETA VARIES
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /TAPE/TAPFIN,TAPEOT,TAPEA,TAPFH,TAPEC,TAPED,TAPEE,TAPEF,
  TAPEG,TAPEH,TAPEI,TAPFJ,TAPFK
1 DIMENSION TITLE(15),AX(100),THETOX(100),THETX(100),DELTHX(100),
  NN(100),SFCT(1),DELZX(100),DELYX(100),AA(100),BB(100)
INTEGER STAT,STATT,STATD,STATC,ERROR
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
  TAPFG,TAPEH,TAPEI,TAPEJ,TAPEK
1 INTEGER STATA, STATR,PAGE,SFG,TYPE,CASE,DISCON
  RADD(BBI,AAI,THP) = SORT(BBI*BBI*COS(THP)*COS(THP) +
  AAI*AAI*SIN(THP)*SIN(THP) )
1 WRITE (TAPFOT,10)
10 FORMAT (1H1,////,1H0,3HELLIPTICAL GEOMETRY DATA IS BEING
  1 21H GENERATED ***** )
C SET COUNTERS
  TYPE = 3
  NRFC = 0
C READ IN TITLE CARD

```

ELLI 001C
 ELLI 002C
 ELLI 003I
 ELLI 004
 ELLI 005
 ELLI 006
 ELLI 007
 ELLI 008
 ELLI 009
 ELLI 010
 ELLI 011
 ELLI 012
 ELLI 013
 ELLI 014
 ELLI 015
 ELLI 016
 ELLI 017
 ELLI 018
 ELLI 019
 ELLI 020
 ELLI 021
 ELLI 022
 ELLI 023
 ELLI 024
 ELLI 025
 ELLI 026
 ELLI 027
 ELLI 028
 ELLI 029
 ELLI 030
 ELLI 031
 ELLI 032
 ELLI 033
 ELLI 034
 ELLI 035

ELLIP

ELLIP

ELLI 036
 ELLI 037
 ELLI 038
 ELLI 039
 ELLI 040
 ELLI 041
 ELLI 042
 ELLI 043
 ELLI 044
 ELLI 045
 ELLI 046
 ELLI 047
 ELLI 048
 ELLI 049
 ELLI 050
 ELLI 051
 ELLI 052
 ELLI 053
 ELLI 054
 ELLI 055
 ELLI 056
 ELLI 057
 ELLI 058
 ELLI 059
 ELLI 060
 ELLI 061
 ELLI 062
 ELLI 063
 ELLI 064
 ELLI 065
 ELLI 066
 ELLI 067
 ELLI 068
 ELLI 069
 ELLI 070

```

20 READ (TAPEIN,30) (TITLE(L),L=1,12),DISCON,I,PRINT,CASE,SECT,ITYPE
30 FORMAT (12A4,11X,2I1,3X12,1A4,12)
IF (ITYPE .NE. 4) GO TO 470
LINE = 100
SEQ = 1
C READ IN ALL DATA CARDS FOR THE SECTION
REWIND TAPEB
I = 1
40 READ (TAPEIN,50) X,THETO,THETL,NN(I),A,B,DFLZ,DELY,LAST,ITYPE
50 FORMAT (F10.0,2F6.0,13,2F10.0,0,2F7.0,11,10X12)
IF (ITYPE .NE. 5) GO TO 470
DELY = (THETL-THETO)/FLOAT(NN(I))
THETO = THETO /57.2957795
THETL = THETL /57.2957795
DELY = DELY /57.2957795
AX(I) = X
THETOX(I) = THETO
THETLX(I) = THETL
DELTHX(I) = DELY
AA(I) = A
BB(I) = B
DELYX(I) = DELY
DELZX(I) = DELZ
IF (LAST .EQ. 0) GO TO 60
N = I
GO TO 70
60 I = I + 1
GO TO 40
70 I = 1
M = 0
80 IF (I .GT. N) GO TO 110
C
IF (NN(I)-M)100,100,90
90 M = NN(I)
100 I = I + 1

```

ELLIP

ELLIP

```

      GO TO 80
C     110 GO TO (120,180,270), DISCON
C     120 N = M + 1
C     DO 170 I=1,N
C     DO 160 J=1,M
C     XA = AX(I)
      THETA = THETX(I) + (FLOAT(J-1)) * DELTAX(I)
      THETAP = ABS(THETA - 1.570796)
      RAD = RADD(BR(I),AA(I),THETAP)
      IF (RAD .NE. 0.0) RAD = AA(I)*BB(I) / RAD
      YA = RAD * SIN(THETA)
      ZA = RAD * COS(THETA)
      YA = YA + DELYX(I)
      ZA = ZA + DELZX(I)
C     IF (J .EQ. 1) GO TO 130
      STATA = 0
      GO TO 150
C     130 IF (I .EQ. 1) GO TO 140
      STATA = 1
      GO TO 150
C     140 STATA = 2
      150 WRITE (TAPER) XA,YA,ZA,STATA
      160 CONTINUE
      170 CONTINUE
C     GO TO 360
C
```

```

ELLI 071
ELLI 072
ELLI 073
ELLI 074
ELLI 075
ELLI 076
ELLI 077
ELLI 078
ELLI 079
ELLI 080
ELLI 081
ELLI 082
ELLI 083
ELLI 084
ELLI 085
ELLI 086
ELLI 087
ELLI 088
ELLI 089
ELLI 090
ELLI 091
ELLI 092
ELLI 093
ELLI 094
ELLI 095
ELLI 096
ELLI 097
ELLI 098
ELLI 099
ELLI 100
ELLI 101
ELLI 102
ELLI 103
ELLI 104
ELLI 105
```

ELLIP

ELLIP

```
C 180 DO 260 I=1,N
C
C     LIM = M+1-NN(I)
C
C     DO 220 J=1,LIM
C     XA = AX(I)
C     THETA = THETX(I)
C     THETA = ABS(THETA - 1.570796)
C     RAD = RADD(BR(J),AA(I),THETA)
C     IF (RAD .NE. 0.0) RAD = AA(I)*BR(I) / RAD
C     YA = RAD * SIN(THETA)
C     ZA = -RAD * COS(THETA)
C     YA = YA + DELYX(I)
C     ZA = ZA + DELZX(I)
C     IF (J .EQ. 1) GO TO 190
C     STATA = 0
C     GO TO 210
C
C 190 IF (I .EQ. 1) GO TO 200
C     STATA = 1
C     GO TO 210
C
C 200 STATA = 2
C 210 WRITE (TAPE8) XA,YA,ZA,STATA
C
C 220 CONTINUE
C     K=0
C     LIM = LIM + 1
C     NM = M + 2
C     DO 250 J = LIM,NM
C     XA = AX(I)
C     THETA = THETX(I) = (FLOAT(NN(I)-K)) * DELTHX(I)
C     THETA = ABS(THETA - 1.570796)
C     RAD = RADD(BR(I),AA(I),THETA)
```

ELLI 106
ELLI 107
ELLI 108
ELLI 109
ELLI 110
ELLI 111
ELLI 112
ELLI 113
ELLI 114
ELLI 115
ELLI 116
ELLI 117
ELLI 118
ELLI 119
ELLI 120
ELLI 121
ELLI 122
ELLI 123
ELLI 124
ELLI 125
ELLI 126
ELLI 127
ELLI 128
ELLI 129
ELLI 130
ELLI 131
ELLI 132
ELLI 133
ELLI 134
ELLI 135
ELLI 136
ELLI 137
ELLI 138
ELLI 139
ELLI 140

ELLIP

ELLIP

ELLI 141
ELLI 142
ELLI 143
ELLI 144
ELLI 145
ELLI 146
ELLI 147
ELLI 148
ELLI 149
ELLI 150
ELLI 151
ELLI 152
ELLI 153
ELLI 154
ELLI 155
ELLI 156
ELLI 157
ELLI 158
ELLI 159
ELLI 160
ELLI 161
ELLI 162
ELLI 163
ELLI 164
ELLI 165
ELLI 166
ELLI 167
ELLI 168
ELLI 169
ELLI 170
ELLI 171
ELLI 172
ELLI 173
ELLI 174
ELLI 175

```
IF (RAD .NE. 0.0) RAD = AA(I)*BB(I) / RAD
YA = RAD * SIN(THETA)
ZA = RAD * COS(THETA)
YA = YA + DELYX(I)
ZA = ZA + DELZX(I)
C
IF (J.EQ. 1) GO TO 230
STAT = 0
GO TO 240
C
230 STAT = 1
C
240 WRITE (TAPEB) XA,YA,ZA,STAT
K = K + 1
250 CONTINUE
C
260 CONTINUE
M = M + 2
GO TO 360
C
270 M = M + 2
DO 350 I = 1,N
NM = NM(I) + 1
C
DO 310 J = 1,NM
XA = AX(I)
THETA = THETOX(I) + (FLOAT(J-1))*DELTHX(I)
THETAP = ABS(THETA - 1.570796)
RAD = RADD(BR(I),AA(I),THEYAP)
IF (RAD .NE. 0.0) RAD = AA(I)*BB(I) / RAD
YA = RAD * SIN(THETA)
ZA = RAD * COS(THETA)
YA = YA + DELYX(I)
ZA = ZA + DELZX(I)
IF (J.EQ.1) GO TO 260
```

ELLIP

ELLIP

ELLI 176
ELLI 177
ELLI 178
ELLI 179
ELLI 180
ELLI 181
ELLI 182
ELLI 183
ELLI 184
ELLI 185
ELLI 186
ELLI 187
ELLI 188
ELLI 189
ELLI 190
ELLI 191
ELLI 192
ELLI 193
ELLI 194
ELLI 195
ELLI 196
ELLI 197
ELLI 198
ELLI 199
ELLI 200
ELLI 201
ELLI 202
ELLI 203
ELLI 204
ELLI 205
ELLI 206
ELLI 207
ELLI 208
ELLI 209
ELLI 210

```

C      STATE = 0
      GO TO 300

C      280 IF (I,EG,1) GO TO 290
      STATE = 1
      GO TO 300

C      290 STATE = 2
      300 WRITE (TAPE8) XA,YA,ZA,STATE

C      310 CONTINUE
      NM = NM+1

C      DO 340 J = NM,M
      XA = AX(I)
      THETA = TMFTLX(I)
      TMFTAP = ABS(THETA - 1.570796)
      RAD = RADQ(BR(I),AA(I),THETA)
      IF (RAD .NE. 0.0) RAD = AA(I)*RB(I) / RAD
      YA = RAD * SIN(THETA)
      ZA = -RAD * COS(THETA)
      YA = YA + DELYX(I)
      ZA = ZA + DELZX(I)
      IF (J,EG,1) GO TO 320
      STATE = 0
      GO TO 330

C      320 STATE = 1
      330 WRITE (TAPE8) XA,YA,ZA,STATE

C      340 CONTINUE

C      350 CONTINUE

C
C
```

ELLIP

ELLIP

```

360 STATA = 3
IF (LAST.EQ.0 .OP. LAST.EQ.2) STATA = 0
C   BACKSPACE TAPEB
    READ (TAPEB) XA,YA,ZA,STAT
    RACKSPACE TAPEB
    WRITE (TAPEB) XA,YA,ZA,STATA
C   REWIND TAPEB
    K = 1
C   370 READ (TAPEB) X,Y,Z,STAT
C   IF ( STAT .GT. 2) GO TO 410
    READ (TAPEB) XX,YY,ZZ,STATT
C   IF ( STATT .GT. 2) GO TO 390
C   380 CALL PUNCH (X,Y,Z,STAT,XX,YY,ZZ,STATT,SECT,TYPE,LINE,SEQ,
1   LAST,IPRINT,NREC)
C   GO TO (370,20,450), K
C
C   390 IF (STATT.EQ. 3) GO TO 400
    STATT = 0
    K = 2
    GO TO 380
C   400 K = 3
    GO TO 380
C
C   410 XR = X
    YH = Y

```

```

ELLI 211
ELLI 212
ELLI 213
ELLI 214
ELLI 215
ELLI 216
ELLI 217
ELLI 218
ELLI 219
ELLI 220
ELLI 221
ELLI 222
ELLI 223
ELLI 224
ELLI 225
ELLI 226
ELLI 227
ELLI 228
ELLI 229
ELLI 230
ELLI 231
ELLI 232
ELLI 233
ELLI 234
ELLI 235
ELLI 236
ELLI 237
ELLI 238
ELLI 239
ELLI 240
ELLI 241
ELLI 242
ELLI 243
ELLI 244
ELLI 245

```

ELLIP

ELLIP

ELLI 246
 ELLI 247
 ELLI 248
 ELLI 249
 ELLI 250
 ELLI 251
 ELLI 252
 ELLI 253
 ELLI 254
 ELLI 255
 ELLI 256
 ELLI 257
 ELLI 258
 ELLI 259
 ELLI 260
 ELLI 261
 ELLI 262
 ELLI 263
 ELLI 264
 ELLI 265
 ELLI 266
 ELLI 267
 ELLI 268
 ELLI 269
 ELLI 270I
 ELLI 271C
 ELLI 272

```

ZB = Z
STATR = STAT
GO TO 430
420 BACKSPACE TAPE
430 BACKSPACE TAPE
READ (TAPEB) XA,YA,ZA,STATA
IF (STATA.EQ.1 .OR. STATA.EQ.2) GO TO 440
GO TO 420
440 STATC = 0
CALL PUNCH (YB,YR,ZB,STATC,XA,YA,ZA,STATA,SECT,TYPE,LINE,SEQ,
1 LAST,IPRINT,NREC)
STATD = 0
IF (STAT.EQ.3) STATD = 3
READ (TAPEB) XC,YC,ZC,STATY
READ (TAPEB) XD,YD,ZD,STATY
CALL PUNCH (XC,YC,ZC,STATC,XD,YD,ZD,STATD,SECT,TYPE,LINE,SEQ,
1 LAST,IPRINT,NREC)
IF (STAT.NE.3) GO TO 20
C
C
450 CONTINUE
460 GO TO 480
470 ERROR = 1
C
480 RETURN
480 CONTINUE
END
  
```

ELLIP

CUBIC

CUBI 001C
 CUBI 002C
 CUBI 003I
 CUBI 004
 CUBI 005
 CUBI 006
 CUBI 007
 CUBI 008
 CUBI 009
 CUBI 010
 CUBI 011
 CUBI 012
 CUBI 013
 CUBI 014
 CUBI 015
 CUBI 016
 CUBI 017
 CUBI 018
 CUBI 019
 CUBI 020
 CUBI 021
 CUBI 022
 CUBI 023
 CUBI 024
 CUBI 025
 CUBI 026
 CUBI 027
 CUBI 028
 CUBI 029
 CUBI 030
 CUBI 031
 CUBI 032
 CUBI 033
 CUBI 034
 CUBI 035

```

OVERLAY (MARK4,1,3)
PROGRAM CUBIC
SURROUTINE CUBIC
C THIS SUBPROGRAM CALCULATES THE QUADRILATERAL DATA FOR A SURFACE GIVEN
C BY THE COONS MIT SURFACE FIT TECHNIQUE.
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF, TAPEG,
      TAPEH, TAPEI, TAPEJ, TAPEK
1 DIMENSION XA(20), XB(20), YA(20), YB(20), ZA(20), ZB(20), XB1(4,20), YB1(
  4,20), ZB1(4,20), NPTS(4), D(4,9), TITLE(15), SECT(1)
REAL L21, L31, L32, L1, M1, M2, N2, LN, MN, NN
INTEGER STAT, STATT, TYPE, SEQ, CASE, PAGE, ERROR
INTEGER TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1
WRITE (TAPEOUT,10)
10 FORMAT (1H://////,1H0,35HPARAMETRIC CURIC GEOMETRY DATA JS
      1 24H BEING GENERATED ***** )
C
      TYPE=3
      SEQ=1
      LINE=100
      NREC=0
C
C *****READ IN BOUNDARY CURVE DATA
C SET UP STARTING CONSTANTS
C
20 CONTINUE
      NZ=1
      LZ=0
30 IIE=1
      ITRUF=0
      IFALSE=1
      READ (TAPEIN,40) (TITLE(K),K=1,12), NQU, NMM, LAST, ISOVR, IPRINT, CASE,
  
```

CUBIC

CUBIC

1 SECT, ITYPE
 40 FORMAT (12A2, 1X, I3, 1X, I3, 1X, I3, 3I1, 2X I2, 1A4, I2)
 IF (ITYPE .NE. 6) GO TO 580
 READ IN BOUNDARY CURVE DATA
 50 CONTINUE
 READ (TAPEIN, 60) X, Y, Z, ISTAT, XX, YY, ZZ, TSTAT, ITYPE
 60 FORMAT(3F10, 4, I1, 3F10, 4, I1, 8X I2)
 IF (ITYPE .NE. 7) GO TO 580
 IFLAG=IFALSE
 GO TO 150
 70 IF(IRFLAG)80, 90, 80
 80 IRFLAG=ITRUE
 XEXX
 YEYY
 Z=ZZ
 ISTAT=ISTAT
 GO TO 100
 90 IRFLAG=IFALSE
 READ (TAPEIN, 60) X, Y, Z, ISTAT, XX, YY, ZZ, TSTAT, ITYPE
 IF (ITYPE .NE. 7) GO TO 580
 100 IF(ISTAT)110, 250, 110
 110 IF(ISTAT=3)120, 250, 120
 120 IF(ISTAT=2)130, 270, 130
 130 IF(IAFLAG=1)140, 270, 140
 140 MCEM
 150 M=1
 IF(ISTAT=2)160, 230, 160
 160 IF(IBFLAG=1)170, 200, 170
 170 DO 180 J=1, MC
 YA(J)=XB(J)
 YB(J)=YA(J)
 180 ZA(J)=ZB(J)
 190 XB(1)=X

C
C
C

CUBIC

CUBIC

CUBI	071
CUBI	072
CUBI	073
CUBI	074
CUBI	075
CUBI	076
CUBI	077
CUBI	078
CUBI	079
CUBI	080
CUBI	081
CUBI	082
CUBI	083
CUBI	084
CUBI	085
CUBI	086
CUBI	087
CUBI	088
CUBI	089
CUBI	090
CUBI	091
CUBI	092
CUBI	093
CUBI	094
CUBI	095
CUBI	096
CUBI	097
CUBI	098
CUBI	099
CUBI	100
CUBI	101
CUBI	102
CUBI	103
CUBI	104
CUBI	105

CUBIC

```

YB(I)=Y
ZB(I)=Z
GO TO 70
200 IF(I AFLAG)210,220,210
210 IFLAG=0
GO TO 170
220 I AFLAG=1
GO TO 190
230 I AFLAG=0
I BFLAG=1
N=N+1
240 YA(M)=X
YA(M)=Y
ZA(M)=Z
GO TO 70
250 N=N+1
IF(I AFLAG)260,240,260
260 XB(M)=X
YB(M)=Y
ZB(M)=Z
IF(I STAT=3)70,270,70
270 ML=MC
MC=M
280 N=N+1
C SET UP ROUND A Y CURVE COORDINATE ARRAYS
290 CONTINUE
IF(I I=1)300,300,320
300 DO 310 I=1,ML
XB(I,I,I)=XA(I)
YB(I,I,I)=YA(I)
ZB(I,I,I)=ZA(I)
NPTS(I,I)=ML
320 II=II+1
DO 330 I=1,MC

```

CUBIC

```

XBI(II,I)=XB(I)
YBI(II,I)=YB(I)
330 ZBI(II,I)=ZB(I)
NPTS(II)=MC
IF(II=4)150,340,340
340 CONTINUE
IF(ISTAT=3)30,350,350
350 CONTINUE
C
C *****CALCULATE BOUNDARY CURVE CONSTANTS
C CALCULATE ARC LENGTH S ON BOUNDARY
NB=1
360 S=0.0
K=NPTS(NR)=2
DO 370 I=2,K
3700S=S+SQRT((XBI(NB,I+1)-XBI(NB,I))**2+(YBI(NB,I+1)-YBI(NB,I))**2+(ZB
1:(NB,I+1)-ZBI(NB,I))**2)
C CALCULATE TANGENT VECTORS AT THE START OF THE BOUNDARY
IFLAG1=0
J1=1
J2=2
J3=3
380 X2X1=XBI(NB,J2)-XBI(NB,J1)
X3X1=XBI(NB,J3)-XBI(NB,J1)
X3X2=XBI(NB,J3)-XBI(NB,J2)
Y2Y1=YBI(NB,J2)-YBI(NB,J1)
Y3Y1=YBI(NB,J3)-YBI(NB,J1)
Y3Y2=YBI(NB,J3)-YBI(NB,J2)
Z2Z1=ZBI(NB,J2)-ZBI(NB,J1)
Z3Z1=ZBI(NB,J3)-ZBI(NB,J1)
Z3Z2=ZBI(NB,J3)-ZBI(NB,J2)
L21=SQRT(X2X1**2+Y2Y1**2+Z2Z1**2)
L31=SQRT(X3X1**2+Y3Y1**2+Z3Z1**2)
L32=SQRT(X3X2**2+Y3Y2**2+Z3Z2**2)

```

CUBI 106
CUBI 107
CUBI 108
CUBI 109
CUBI 110
CUBI 111
CUBI 112
CUBI 113
CUBI 114
CUBI 115
CUBI 116
CUBI 117
CUBI 118
CUBI 119
CUBI 120
CUBI 121
CUBI 122
CUBI 123
CUBI 124
CUBI 125
CUBI 126
CUBI 127
CUBI 128
CUBI 129
CUBI 130
CUBI 131
CUBI 132
CUBI 133
CUBI 134
CUBI 135
CUBI 136
CUBI 137
CUBI 138
CUBI 139
CUBI 140

CUBIC

CUBIC

CUBI 141
CUBI 142
CUBI 143
CUBI 144
CUBI 145
CUBI 146
CUBI 147
CUBI 148
CUBI 149
CUBI 150
CUBI 151
CUBI 152
CUBI 153
CUBI 154
CUBI 155
CUBI 156
CUBI 157
CUBI 158
CUBI 159
CUBI 160
CUBI 161
CUBI 162
CUBI 163
CUBI 164
CUBI 165
CUBI 166
CUBI 167
CUBI 168
CUBI 169
CUBI 170
CUBI 171
CUBI 172
CUBI 173
CUBI 174
CUBI 175

```
L1=X3X1/L31
M1=Y3Y1/L31
N1=Z3Z1/L31
L2=X2X1/L21
M2=Y2Y1/L21
N2=Z2Z1/L21
LN=(N1*(L1*N2=L2*N1)+M1*(L1*N2=L2*M1))
MN=(M1*(M1*N2=M2*N1)+L1*(L1*N2=L2*M1))
NN=N1*(M1*N2=M2*N1)+L1*(L1*N2=L2*N1)
COSEP1=(X2X1*X3X1+Y2Y1*Y3Y1+Z2Z1*Z3Z1)/(L21*L31)
IF(COSEP1=0,999999)400,400,390
390 EPS1=0.0
GO TO 430
400 IF(COSFP1+0,999999)410,420,420
410 EPS1=0.0
GO TO 430
420 EPS1=ARCOS(CNSEP1)
430 COSEP2=(X3X2*X3X1+Y3Y2*Y3Y1+Z3Z2*Z3Z1)/(L32*L31)
440 EPS2=0.0
GO TO 480
450 IF(COSEP2+0,999999)460,470,470
460 EPS2=0.0
GO TO 480
470 EPS2=ARCOS(CNSFP2)
480 DELTA=EPS1+EPS2
TX=L1*COS(DELTA)+LN*SIN(DELTA)
TY=M1*COS(DELTA)+MN*SIN(DELTA)
TZ=N1*COS(DELTA)+NN*SIN(DELTA)
C CALCULATE END POINT DERIVATIVES
IF(IFLAG1)490,490,500
490 X1VON=TX*S
Y1VON=TY*S
Z1VON=TZ*S
```

CUBIC

CUBIC

CUBI 176
 CUBI 177
 CUBI 178
 CUBI 179
 CUBI 180
 CUBI 181
 CUBI 182
 CUBI 183
 CUBI 184
 CUBI 185
 CUBI 186
 CUBI 187
 CUBI 188
 CUBI 189
 CUBI 190
 CUBI 191
 CUBI 192
 CUBI 193
 CUBI 194
 CUBI 195
 CUBI 196
 CUBI 197
 CUBI 198
 CUBI 199
 CUBI 200
 CUBI 201
 CUBI 202
 CUBI 203
 CUBI 204
 CUBI 205
 CUBI 206
 CUBI 207
 CUBI 208
 CUBI 209
 CUBI 210

CUBIC

```

J1=NPTS(NB)-2
J2=NPTS(NB)-1
J3=NPTS(NB)
IFLAG1=1
GO TO 380
500 X1V01=TX+S
    Y1V01=TY+S
    Z1V01=TZ+S
C
C *****CALCULATE CONSTANTS FOR BOUNDARY CURVE
D(NB,1)=2.0*(XB1(NB,2)-XB1(NB,J2))+X1V00+X1V01
D(NB,2)=3.0*(X61(NB,J2)-XB1(NB,2))-2.0*X1V00-X1V01
D(NB,3)=X1V00
D(NB,4)=2.0*(YB1(NB,2)-YB1(NB,J2))+Y1V00+Y1V01
D(NB,5)=3.0*(YB1(NB,J2)-YB1(NB,2))-2.0*Y1V00-Y1V01
D(NB,6)=Y1V00
D(NB,7)=2.0*(ZB1(NB,2)-ZB1(NB,J2))+Z1V00+Z1V01
D(NB,8)=3.0*(ZB1(NB,J2)-ZB1(NB,2))-2.0*Z1V00-Z1V01
D(NB,9)=Z1V00
NB=NR+1
IF(NB=4)360,360,510
C
C *****CALCULATE PATCH DATA
510 N0W=N0W/2+1
    DELU=1.0/FLOAT(N0W)
    DELW=1.0/FLOAT(N0W)
    N0U=N0U+1
    N0W=N0W+1
    STATT=0
    US0=0
C
DO 560 I=1,N0U
STAT=1
INUBC
  
```

CUBIC

CUBI 211
 CUBI 212
 CUBI 213
 CUBI 214
 CUBI 215
 CUBI 216
 CUBI 217
 CUBI 218
 CUBI 219
 CUBI 220
 CUBI 221
 CUBI 222
 CUBI 223
 CUBI 224
 CUBI 225
 CUBI 226
 CUBI 227
 CUBI 228
 CUBI 229
 CUBI 230
 CUBI 231
 CUBI 232
 CUBI 233
 CUBI 234
 CUBI 235
 CUBI 236
 CUBI 237
 CUBI 238
 CUBI 239
 CUBI 240
 CUBI 241
 CUBI 242
 CUBI 243
 CUBI 244
 CUBI 245

```

C      W=0.0
C      DO 550 K=1,NPW
C
C      W3=W**3
C      W2=W**2
C      U3=U**3
C      U2=U**2
C      CALCULATE BLENDING FUNCTIONS
C      F11=3.0*U2+2.0*U3
C      F0U=1.0-3.0*U2+2.0*U3
C      F1W=3.0*W2+2.0*W3
C      F0W=1.0-3.0*W2+2.0*W3
C
C      CALCULATE POINTS ON BOUNDARY CURVES
C      X0W=D(1,1)*W3+D(1,2)*W2+D(1,3)*W+XB1(1,2)
C      Y0W=D(1,4)*W3+D(1,5)*W2+D(1,6)*W+YB1(1,2)
C      Z0W=D(1,7)*W3+D(1,8)*W2+D(1,9)*W+ZB1(1,2)
C      X1W=D(2,1)*W3+D(2,2)*W2+D(2,3)*W+XB1(2,2)
C      Y1W=D(2,4)*W3+D(2,5)*W2+D(2,6)*W+YB1(2,2)
C      Z1W=D(2,7)*W3+D(2,8)*W2+D(2,9)*W+ZB1(2,2)
C      XU=D(3,1)*U3+D(3,2)*U2+D(3,3)*U+XB1(3,2)
C      YU=D(3,4)*U3+D(3,5)*U2+D(3,6)*U+YB1(3,2)
C      ZU=D(3,7)*U3+D(3,8)*U2+D(3,9)*U+ZB1(3,2)
C      XU1=D(4,1)*U3+D(4,2)*U2+D(4,3)*U+XB1(4,2)
C      YU1=D(4,4)*U3+D(4,5)*U2+D(4,6)*U+YB1(4,2)
C      ZU1=D(4,7)*U3+D(4,8)*U2+D(4,9)*U+ZB1(4,2)
C      NPT1=NPTS(1)-1
C      NPT2=NPTS(2)-1
C
C      CALCULATE POSITION OF A POINT ON THE SURFACE
C      CXS = X0W*F0U+X1W*F1U+XU0*F0W+XU1*F1W+XU1*F1W+XU1*F1W*(1,NPT1)*
C      1 F0U*F1W+XB1(2,2)*F1U+Y0W*F0U+YU0*F0W+YU1*F1W+YU1*F1W*(1,NPT1)*
C      OYS = Y0W*F0U+Y1W*F1U+YU0*F0W+YU1*F1W+YU1*F1W*(1,NPT1)*
  
```

CUBIC

CUBIC

CUBI 246
 CUBI 247
 CUBI 248
 CUBI 249
 CUBI 250
 CUBI 251
 CUBI 252
 CUBI 253
 CUBI 254
 CUBI 255
 CUBI 256
 CUBI 257
 CUBI 258
 CUBI 259
 CUBI 260
 CUBI 261
 CUBI 262
 CUBI 263
 CUBI 264
 CUBI 265
 CUBI 266
 CUBI 267
 CUBI 268
 CUBI 269
 CUBI 270
 CUBI 271
 CUBI 272
 CUBI 273
 CUBI 274

```

1  FOU*F1W=YR1(2,2)*F1U*FOW=YR1(2,NPT2)*F1U*F1W
0ZS = ZOW*FOU+Z1W*F1U+ZUO*FOW+ZU1*F1W+ZR1(1,2)*FOU*FOW-ZB1(1,NPT1)*
1  FOU*F1W-ZB1(2,2)*F1U*FOW-ZR1(2,NPT2)*F1U*F1W
C
IF(INU=1)520,530,530
520 XYS = XS
    YYS = YS
    ZS = ZS
    INU=1
    GO TO 540
530 IF (I,EG,NOU .AND. K,EG,NOW .AND. LAST,EG,1) STATT = 3
    IF (I,EG,NOU .AND. K,EG,NOW .AND. LAST,EG,3) STATT = 3
    IF (I,EG,NOU .AND. K,EG,NOW .AND. LAST,EG,4) STATT = 3
    IF (I,EG,1 .AND. K,EG,2 .AND. ISOVR,EG,0) STAT = 2
    CALL PUNCH (XYS,YYS,ZS,STAT,XS,YS,ZS,STATT,SECT,TYPE,LINE,SEG,
1  LAST,IPRINT,NREC)
    INU=0
    STAT=0
540 W=W+DELU
550 CONTINUE
    U=U+DELU
560 CONTINUE
C
IF (STATT .NE. 3) GO TO 20
570 GO TO 590
580 ERROR = 1
C 590 RETURN
    END
  
```

CUBIC

AIRCFT

AIRC 001C
 AIRC 002C
 AIRC 003I
 AIRC 004
 AIRC 005
 AIRC 006
 AIRC 007
 AIRC 008
 AIRC 009
 AIRC 010
 AIRC 011
 AIRC 012
 AIRC 013
 AIRC 014
 AIRC 015
 AIRC 016
 AIRC 017
 AIRC 018
 AIRC 019
 AIRC 020
 AIRC 021
 AIRC 022
 AIRC 023
 AIRC 024
 AIRC 025
 AIRC 026
 AIRC 027
 AIRC 028
 AIRC 029
 AIRC 030
 AIRC 031
 AIRC 032
 AIRC 033
 AIRC 034
 AIRC 035

OVERLAY (MARK4,1,4)
 PROGRAM AIRCFT
 SUBROUTINE AIRCFT

 *****MAIN PROGRAM FOR AIRCRAFT TYPE VEHICLE GEOMETRY GENERATION *****
 *****OPTION, THIS OPTION MAY ALSO BE USED TO CONVERT GEOMETRY *****
 *****DATA FOR THE NASA (HARRIS) WAVE DRAG PROGRAM TO ELEMENT *****
 *****DATA FORMAT FOR USE BY THE HYPERSONIC ARBITRARY-BODY *****
 *****PROGRAM OR BY THE ARBITRARY-BODY GRAPHICS PROGRAMS. *****

*** THIS PROGRAM WRITTEN BY A.E. GENTRY AND D.N. SMYTH ***
 DOUGLAS AERO RESEARCH GROUP, LONG BEACH

NOTE- THE INPUT DATA MUST BE EXACTLY AS REQUIRED FOR THE
 CONFIGURATION DATA FOR THE HARRIS PROGRAM (INCLUDING
 THE IDENTIFICATION CARD). MULTIPLE CONFIGURATIONS
 CAN BE STACKED ONE BEHIND THE OTHER.

DIMENSION

1 XAF(30),WAFORG(20,4),IZORD(20,30),WAFORD(20,30),
 2 NRADX(4),NFORX(4),XFUS(4,30),ZFUS(4,30),FUSARD(4,30),
 3 YFUSY(4,30,30),ZFUSZ(4,30,30),
 4 PNDORG(9,6),XPDD(9,30),PNDP(9,30),TOR(9),IEL(9),NEL(9),
 5 FINGRG(6,2,4),XFIN(6,10),FINORD(6,10),
 6 CANORG(2,2,4),XCAN(2,10),CANORD(2,10),CANORI(2,10),
 7 XOC(30),AFORG(20,7),AFCAM(20,30),AFORD(20,30),
 8 CARD(21),SURF2(6),SURF3(2)

COMMON /ACFT/IPRINT,ITAPF,ITYPE,PI,DX,CASE

COMMON /ARFOIL/ NAF,NAFORD,NCAM,NACA,ITYP,IRONT,
 XOC,AFORG,AFCAM,AFORD,SURFID

AIRCFT

AIRCFT

```

2      , ISURF, IFLAP, NHNGE
C
COMMON /FUSD/J2,J6,NFUS,NRADX,NFORX,XFUS,ZFUS,YFUSY,7FUSZ,FUSARD
C
COMMON /NACELD/NP,NPOOR,PODRG,XPOD,POOR,IOR,IEL,NEL
C
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPFB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /GFLAG/IOUT,ISTAT,IORIEN,COMPIN
C
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPFG,TAPEH,TAPEI,TAPFJ,TAPEK
INTEGER CASE
C
DATA TYCH/2H99/
DATA SURF1/4HWING/,
1 SURF2(1),SURF2(2),SURF2(3),SURF2(4),SURF2(5),SURF2(6)/
2 4HFIN1,4HFIN2,4HFIN3,4HFIN4,4HFIN5,4HFIN6/,
3 SURF3(1),SURF3(2)/4MCAN1,4MCAN2/
IPRINT = 1
C
ITAPE = IOUT
ITYPE = 3
PI = 3.1415927
C
C**PROGRAM OPERATION STARTS HERE***
C
C HEAD IDENTIFICATION CARD, CONTROL CARD, AND REFERENCE AREA
C CARD (IF REQUIRED)
10 READ (TAPEIN,20) (CARD(I),I=1,17),ISTAT3,IMARIS, CASE,
1 (CARD(I), I=19,21)
20 FORMAT (9A3,6A4,I1,2X I1,1X I2,4XA2,2A4)
IF (CARD(19).EQ. TYCH) GO TO 780
C CHECK FOR NASA-HARRIS INPUT COORDINATES.

```

```

AIRC 036
AIRC 037
AIRC 038
AIRC 039
AIRC 040
AIRC 041
AIRC 042
AIRC 043
AIRC 044
AIRC 045
AIRC 046
AIRC 047
AIRC 048
AIRC 049
AIRC 050
AIRC 051
AIRC 052
AIRC 053
AIRC 054
AIRC 055
AIRC 056
AIRC 057
AIRC 058
AIRC 059
AIRC 060
AIRC 061
AIRC 062
AIRC 063
AIRC 064
AIRC 065
AIRC 066
AIRC 067
AIRC 068
AIRC 069
AIRC 070

```

AIRCFT

AIRCFT

AIRC 071
AIRC 072
AIRC 073
AIRC 074
AIRC 075
AIRC 076
AIRC 077
AIRC 078
AIRC 079
AIRC 080
AIRC 081
AIRC 082
AIRC 083
AIRC 084
AIRC 085
AIRC 086
AIRC 087
AIRC 088
AIRC 089
AIRC 090
AIRC 091
AIRC 092
AIRC 093
AIRC 094
AIRC 095
AIRC 096
AIRC 097
AIRC 098
AIRC 099
AIRC 100
AIRC 101
AIRC 102
AIRC 103
AIRC 104
AIRC 105

```

DX = 1.0
IF (IHARIS .EG. 1) DX = -1.0

WRITE (TAPEOT,30) (CARD(I),I=1,17), CASE
30 FORMAT (1H1,48H*** OUTPUT GEOMETRY DATA FROM AIRCRAFT GEOMETRY ,
11HOPTION ***/1H0,5X,13HCONFIGURATION,5X,9A3,8A4,6X,5HCASE ,I2,
2 //,6X,1HX,9X,1HY,9X,1HZ,4X,1HS,5X,1HX,9X,1HY,9X,1HZ,4X,1HS,/1H0)
READ (TAPEIN,40) J0,J1,J2,J3,J4,J5,J6,J7,NWAF,NWAFOR,NFUS,
1 (NRADX(I),NFORX(I),I=1,4),NP,NPODDR,NF,NFINOR,NEAN,NCANOR
40 FORMAT (7I3,I1,I2,16I3,8X)
IF (J0 .EG. 1) PEAD (TAPEIN,50) REFA
50 FORMAT (F7.2,73X)

C
C***CONTROL FLAG CARD DATA CHECK
IF(IABS(J1) .NE. 1) GO TO 60
I = 0
IF (NWAF .LT. 2 .OR. NWAF .GT. 20) GO TO 750
I = 10
IF (NWAFOR .LT. 3 .OR. NWAFOR .GT. 30) GO TO 750
60 IF(IABS(J2) .NE. 1) GO TO 80
I = 11
IF (NFUS .LT. 1 .OR. NFUS .GT. 4) GO TO 750
DO 70 J = 1,NFUS
I = I + 1
IF (NRADX(J) .LT. 3 .OR. NRADX(J) .GT. 30) GO TO 750
I = I + 1
IF (NFORX(J) .LT. 2 .OR. NFORX(J) .GT. 30) GO TO 750
70 CONTINUE
80 IF (J3 .NE. 1) GO TO 90
I = 20
IF (NP .LT. 1 .OR. NP .GT. 9) GO TO 750
I = 21
IF (NPODDR .LT. 2 .OR. NPODDR .GT. 30) GO TO 750
90 IF (J4 .NE. 1) GO TO 100
I = 22

```

AIRCFT

AIRCFT

AIRC 106
AIRC 107
AIRC 108
AIRC 109
AIRC 110
AIRC 111
AIRC 112
AIRC 113
AIRC 114
AIRC 115
AIRC 116
AIRC 117
AIRC 118
AIRC 119
AIRC 120
AIRC 121
AIRC 122
AIRC 123
AIRC 124
AIRC 125
AIRC 126
AIRC 127
AIRC 128
AIRC 129
AIRC 130
AIRC 131
AIRC 132
AIRC 133
AIRC 134
AIRC 135
AIRC 136
AIRC 137
AIRC 138
AIRC 139
AIRC 140

```
IF (NF .LT. 1 .OR. NF .GT. 6) GO TO 750
I = 23
IF (NFINOR .LT. 3 .OR. NFINOR .GT. 10) GO TO 750
100 IF (J5 .NE. 1) GO TO 110
I = 24
IF (NCAN .LT. 1 .OR. NCAN .GT. 2) GO TO 750
I = 25
N = IABS(NCANOR)
IF (N .LT. 3 .OR. N .GT. 10) GO TO 750
110 CONTINUE
C
C**END OF CONTROL FLAG DATA CHECK
C
C
C
C**ENDING**
C*****
IF (J1.EQ.0 .OR. J1.EQ.2) GO TO 260
READ PERCENT-CHORD LOCATIONS
READ (TAPEIN,120) (XAF(I),I=1,NWAFOR)
120 FORMAT (10F7.0,10X)
C READ LOCATION OF CHORD LEADING EDGES AND LENGTH
DO 140 I=1,NWAF
READ (TAPEIN,130) (WAFOR(I,J),J=1,4)
130 FORMAT (4F7.0,52X)
140 CONTINUE
C READ IN CAMBER LINE DATA
NCAM = 0
IF (J1 .EQ. =1) GO TO 170
NCAM = 1
DO 160 J=1,NWAF
READ (TAPEIN,150) (TZORU(J,I),I=1,NWAFOR)
150 FORMAT (10F7.0,10X)
160 CONTINUE
GO TO 200
```

AIRCFT

AIRCFT

```
C CAMBER IS NOT INPUT, ZERO OUT VALUES
170 DO 190 J=1,NWAF
    DO 180 I=1,NWAFOR
        TZORD(J,I) = 0.0
180 CONTINUE
190 CONTINUE
C
C READ IN AIRFOIL ORDINATE DATA
200 DO 210 J=1,NWAF
    READ (TAPEIN,150) (*AFORD(J,I),I=1,NWAFOR)
210 CONTINUE
C
C INITIALIZE DATA FOR WING SURFACE GENERATION
NAF = NWAF
NAFORD = NWAFOR
SURFID = SURFI
DO 240 I = 1,NAF
    DO 220 J = 1,3
        AFORG(I,J) = WAFORG(I,J)
        AFORG(I,4) = AFORG(I,1) - WAFORG(I,4)*DX
        AFORG(I,5) = WAFORG(I,2)
        AFORG(I,6) = WAFORG(I,3)
        AFORG(I,7) = 0.0
C
    DO 230 K = 1,NAFORD
        AFCAM(I,K) = TZORD(I,K)*100./WAFORG(I,4)
230 AFRD(I,K) = WAFORD(I,K)
240 CONTINUE
    DO 250 K = 1,NAFORD
        XNF(K) = XAF(K)
C
        NACA = 0
        ITTP = 0
        TRONT = 0
```

AIRCFT

AIRCFT

AIRC 176
AIRC 177
AIRC 178
AIRC 179
AIRC 180
AIRC 181
AIRC 182
AIRC 183
AIRC 184
AIRC 185
AIRC 186
AIRC 187
AIRC 188
AIRC 189
AIRC 190
AIRC 191
AIRC 192
AIRC 193
AIRC 194
AIRC 195
AIRC 196
AIRC 197
AIRC 198
AIRC 199
AIRC 200
AIRC 201
AIRC 202
AIRC 203
AIRC 204
AIRC 205
AIRC 206
AIRC 207
AIRC 208
AIRC 209
AIRC 210

AIRCFT

```

C      TSURF = 0
C      IFLAP = 0
C      CALL SURF
C
C**FUSFLAG**
C*****
C      260 IF (J2.EQ.0 .OR. J2.FO.2) GO TO 360
C
C      START CYCLE ON FUSELAGE SEGMENTS (NFUS TIMES)
C      GO 350 I=1,NFUS
C
C      NFOX = NFORX(I)
C      NRAY = NRADX(I)
C      READ X-VALUES OF FUSELAGE STATIONS FOR A SEGMENT
C      READ (TAPFIN,270) (XFUS(I,J),J=1,NFOX)
C      270 FORMAT (10F7.0,10X)
C
C      CHECK CIRCULAR AND CAMBER FLAGS
C      IF (J2.EQ.1 .AND. J6.NE.2) GO TO 300
C
C      ZERO OUT CAMBER WHEN NOT REQUIRED
C      280 DO 290 J=1,NFOX
C      290 ZFUS(I,J) = 0.0
C      IF (J2.EQ.1) GO TO 310
C      IF (J2.EQ.1 .AND. J6.NE.0) GO TO 330
C      IF (J2.EQ.1 .AND. J6.EQ.1) GO TO 330
C
C      READ FUSELAGE CAMBER DATA
C      300 READ (TAPFIN,270) (ZFUS(I,J),J=1,NFOX)
C      IF (J2.EQ.-1) GO TO 330
C
C      READ IN FUSELAGE CROSS-SECTION COORDINATES
C      310 DO 320 J=1,NFOX
```

AIRCFT

AIRC 211
AIRC 212
AIRC 213
AIRC 214
AIRC 215
AIRC 216
AIRC 217
AIRC 218
AIRC 219
AIRC 220
AIRC 221
AIRC 222
AIRC 223
AIRC 224
AIRC 225
AIRC 226
AIRC 227
AIRC 228
AIRC 229
AIRC 230
AIRC 231
AIRC 232
AIRC 233
AIRC 234
AIRC 235
AIRC 236
AIRC 237
AIRC 238
AIRC 239
AIRC 240
AIRC 241
AIRC 242
AIRC 243
AIRC 244
AIRC 245

```

      READ (TAPEIN,270) (YFUSY(I,J,N),N=1,NRAX)
      READ (TAPEIN,270) (ZFUSZ(I,J,N),N=1,NRAX)
320  CONTINUE
      GO TO 340
C
C      READ IN FUSELAGE AREAS WHEN CIRCULAR
330  READ (TAPEIN,270) (FUSARJ(I,J),J=1,NFOY)
C
C      340 CONTINUE
C      END OF FUSELAGE SFGMENT DO-LOOP
350  CONTINUE
C
C      CALL FUSELAGE SUBROUTINE
      CALL FUSE
C
C      IFLAP = 0
C
C**POD**
C*****
360  IF (J3.EQ.0 .OR. J3.EQ.2) GO TO 400
C
      DO 390 I=1,NP
C      READ POD ORIGIN DATA
      READ (TAPEIN,370) (PDDORG(I,J),J=1,5)
      I ,IOR(I),(PDDORG(I,J),J=4,6),TEL(I),NFL(I)
370  FORMAT(3F7.0,3X11,3X3F7.0,6X11,5X12)
C      READ X-COORDINATES FOR POD RADII
      READ (TAPEIN,380) (XPDD(I,J),J=1,NPDDOR)
380  FORMAT(10F7.0,10X)
C      READ POD-RADII DISTRIBUTION
      READ (TAPEIN,390) (PDDR(I,J),J=1,NPDDOR)
390  CONTINUE
C
C      CALL POD SUBROUTINE
```

AIRCFT

AIRCFT

AIRC 246
AIRC 247
AIRC 248
AIRC 249
AIRC 250
AIRC 251
AIRC 252
AIRC 253
AIRC 254
AIRC 255
AIRC 256
AIRC 257
AIRC 258
AIRC 259
AIRC 260
AIRC 261
AIRC 262
AIRC 263
AIRC 264
AIRC 265
AIRC 266
AIRC 267
AIRC 268
AIRC 269
AIRC 270
AIRC 271
AIRC 272
AIRC 273
AIRC 274
AIRC 275
AIRC 276
AIRC 277
AIRC 278
AIRC 279
AIRC 280

```
C      CALL NACEL
C
C      TFIAP = 0
C
C**FIN**
C*****
C      GOO IF (J4.EQ.0 .OR. J4.EQ.2) GO TO 490
C
C      NAF = 2
C      NAFORD = NFINOR
C      NCAM = 0
C      NACA = 0
C      ITTP = 0
C      IRNOT = 0
C      TSURF = 0
C      TFIAP = 0
C      READ FIN DATA (NF FINS)
C      DO 430 NNE1,NF
C
C      READ ORDINATE DATA AND LENGTH
C      READ (TAPEIN,410) ((FINORG(NN,I),J),J=1,4),I=1,2)
C      GO 410 FOPVAT (8F7.0,24X)
C      READ PERCENT=CHORD LOCATIONS
C      READ (TAPEIN,420) (XFIN(NN,I),I=1,NFINOR)
C      GO 420 FOPVAT (10F7.0,10X)
C      READ FIN AIRFOIL ORDINATES
C      READ (TAPEIN,420) (FINORD(NN,I),I=1,NFINOR)
C
C      430 CONTINUE
C
C      INITIALIZE DATA FOR FIN SURFACE GENERATION.
C      DO 440 NN = 1,NF
C      SURFTD = SURF2(NN)
C      DO 460 I = 1,NAF
C      DO 440 J = 1,3
```

AIRCFT

AIRCFT

AIRC 281
AIRC 282
AIRC 283
AIRC 284
AIRC 285
AIRC 286
AIRC 287
AIRC 288
AIRC 289
AIRC 290
AIRC 291
AIRC 292
AIRC 293
AIRC 294
AIRC 295
AIRC 296
AIRC 297
AIRC 298
AIRC 299
AIRC 300
AIRC 301
AIRC 302
AIRC 303
AIRC 304
AIRC 305
AIRC 306
AIRC 307
AIRC 308
AIRC 309
AIRC 310
AIRC 311
AIRC 312
AIRC 313
AIRC 314
AIRC 315

```
440 AFORG(I,J) = FINORG(NN,I,J)
   AFORG(I,4) = AFORG(I,1) - FINORG(NN,I,4)*DX
   AFORG(I,5) = AFORG(I,2)
   AFORG(I,6) = AFORG(I,3)
   AFORG(I,7) = 90.0
C
   DO 450 K = 1,NAFORD
   AFCAM(I,K) = 0.0
450 AFORD(I,K) = FINORD(NN,K)
460 CONTINUE
   DO 470 K = 1,NAFORD
470 XOC(K) = XFIN(NN,K)
C
C   CALL SURF
480 CONTINUE
C   FND OF FIN DO LOOP
C
C**CANARD OR HORIZONTAL TAIL
C*****
490 IF (J5.EQ.0 .OR. J5.EQ.2) GO TO 600
C
   N = IABS(NCANOR)
   NAF = 2
   NAFORD = N
   NACA = 0
   ITIP = 0
   IRONY = 0
   ISURF = 0
   IFLAP = 0
C   READ IN ALL CANARD DATA (NCAN CANARDS)
   DO 540 NN = 1,NCAN
C   READ ORIGINATE DATA AND LENGTH
   READ (TAPEIN,500) ((CANORG(NN,I,J),J=1,4),I=1,2)
```

AIRCFT

AIRCFT

AIRC 316
AIRC 317
AIRC 318
AIRC 319
AIRC 320
AIRC 321
AIRC 322
AIRC 323
AIRC 324
AIRC 325
AIRC 326
AIRC 327
AIRC 328
AIRC 329
AIRC 330
AIRC 331
AIRC 332
AIRC 333
AIRC 334
AIRC 335
AIRC 336
AIRC 337
AIRC 338
AIRC 339
AIRC 340
AIRC 341
AIRC 342
AIRC 343
AIRC 344
AIRC 345
AIRC 346
AIRC 347
AIRC 348
AIRC 349
AIRC 350

```
500 FORMAT (BET,0,24X)
C   READ PERCENT-CHORD LOCATIONS
   READ (TAPEIN,510) (XCAN(NN,J),J=1,N)
510 FORMAT (10F7.0,10X)
C   READ FIN AIRFOIL ORDINATES
   READ (TAPEIN,510) (CANORD(NN,J),J=1,N)
C
C   IF AIRFOIL IS NOT SYMMETRICAL READ LOWER ORDINATES
   NCAW = 0
   IF (NCAW .GT. 0) GO TO 520
   NCAW = 1
   READ (TAPEIN,510) (CANORI(NN,J),J=1,N)
   GO TO 540
C
C   SET LOWER SURFACE EQUAL TO UPPER FOR SYMMETRICAL
520 DO 530 J=1,N
530 CANORI(NN,J) = CANORD(NN,J)
C
540 CONTINUE
C
C   INITIALIZE DATA FOR CAMARD SURFACE GENERATION.
   DO 550 NN = 1,NCAN
   SUPFI = SURF3(NN)
   DO 570 I = 1,NAF
   DO 550 J = 1,3
   AFORG(I,J) = CANORG(NN,I,J)
   AFORG(I,4) = AFORG(I,1) - CANORG(NN,I,4)*DX
   AFORG(I,5) = AFORG(I,2)
   AFORG(I,6) = AFORG(I,3)
   AFORG(I,7) = 0.0
C
   DO 560 K = 1,NAFORD
   AFORD(I,K) = (CANORD(NN,K) + CANORI(NN,K))*0.5
560 AFCAW(I,K) = CANORD(NN,K) - AFORG(I,K)
570 CONTINUE
```

AIRCFT

AIRCFT

```

      DO 580 K = 1, NAFORD
      580 XOC(K) = XCAN(NN,K)
C
C      CALL SURF
      590 CONTINUE
C      END OF CANARD DO LOOP.
C
C
C**ARBITRARY AIRFOIL SURFACE
C*****
      600 IF (J7 .NE. 1) GO TO 740
C
C      READ CONTROL CARD
      610 READ (YAPEIN,620) ISURF,NAF,NAFORD,NCAM,NACA,ITIP,IPOGT,ISIMC,
      1
C      620 FORMAT (22X13, 2(3X12), 7(4X11),2X44,4X)
C
C**      CHECK SURFACE TYPE, ISURF = 1(UPPER ONLY), 2(LOWER ONLY),
C
C      IFLAP = 0
      NHWGE = IABS(ISURF)
      IF (NHWGE .LE. 1) GO TO 630
      ARS = GT. 1, (WITH CONTROL SURFACE).
C
C      CONTROL SURFACE INPUT INDICATED. CHECK IF CONSTRAINTS MET.
C      NAFORD MUST BE ODD AND NHWGE EVEN. IF THESE CONDITIONS ARE
C      NOT SATISFIED, FLAP IS NEGLECTED.
C
C      ISURF = ISURF/NHWGE
      IF ((NAFORD/2 + NAFORD/2) .NE. (NAFORD .. 1)) GO TO 630
      IF ((NHWGE/2 + NHWGE/2) .NE. NHWGE) GO TO 630
C
C**      CONSTRAINTS SATISFIED.

```

AIRC 351
AIRC 352
AIRC 353
AIRC 354
AIRC 355
AIRC 356
AIRC 357
AIRC 358
AIRC 359
AIRC 360
AIRC 361
AIRC 362
AIRC 363
AIRC 364
AIRC 365
AIRC 366
AIRC 367
AIRC 368
AIRC 369
AIRC 370
AIRC 371
AIRC 372
AIRC 373
AIRC 374
AIRC 375
AIRC 376
AIRC 377
AIRC 378
AIRC 379
AIRC 380
AIRC 381
AIRC 382
AIRC 383
AIRC 384
AIRC 385

AIRCFT

AIRCFT

AIRC	386
AIRC	387
AIRC	388
AIRC	389
AIRC	390
AIRC	391
AIRC	392
AIRC	393
AIRC	394
AIRC	395
AIRC	396
AIRC	397
AIRC	398
AIRC	399
AIRC	400
AIRC	401
AIRC	402
AIRC	403
AIRC	404
AIRC	405
AIRC	406
AIRC	407
AIRC	408
AIRC	409
AIRC	410
AIRC	411
AIRC	412
AIRC	413
AIRC	414
AIRC	415
AIRC	416
AIRC	417
AIRC	418
AIRC	419
AIRC	420

AIRCFT

```

IFLAP = 1
630 CONTINUE
C READ CHORD LOCATIONS
  READ (TAPEIN,640) (XOC(J),J=1,NAFORD)
640 FORMAT(10F7.0,10X)
C
C READ AIRFOIL SECTION ORIENTATION DATA
  DO 650 I = 1,NAF
  READ (TAPEIN,640) (AFORG(I,J),J=1,7)
650 CONTINUE
C
C CHECK FOR CAMBER DATA
  IF (NCAM .NE. 1) GO TO 680
  IN = NAF
  IF (ISIMC .EQ. 1) IN = 1
  DO 660 I = 1,IN
  READ (TAPEIN,640) (AFCAM(I,J),J=1,NAFORD)
  IF (IN .EQ. NAF) GO TO 700
  DO 670 I = 2,NAF
  DO 670 J = 1,NAFORD
  AFCAM(I,J) = AFCAM(1,J)
  GO TO 700
C NO CAMBER DATA INPUT, SET TO ZERO.
680 DO 690 I = 1,NAF
  DO 690 J = 1,NAFORD
  AFCAM(I,J) = 0.0
C
C 700 CONTINUE
C READ IN THICKNESS DISTRIBUTION
  IN = NAF
  IF (ISIMT .EQ. 1) IN = 1
  DO 710 I = 1,IN
  READ (TAPEIN,640) (AFORD(I,J),J=1,NAFORD)
  IF (IN .EQ. NAF) GO TO 730

```

AIRCFT

AIRC	421
AIRC	422
AIRC	423
AIRC	424
AIRC	425
AIRC	426
AIRC	427
AIRC	428
AIRC	429
AIRC	430
AIRC	431
AIRC	432
AIRC	433
AIRC	434
AIRC	435
AIRC	436
AIRC	437
AIRC	438
AIRC	439
AIRC	440
AIRC	441
AIRC	442
AIRC	443
AIRC	444
AIRC	445
AIRC	446
AIRC	447

AIRCFT

```

DO 720 I = 2, NAF
DO 720 J = 1, NAFORD
720 AFORD(I,J) = AFORD(1,J)
C
730 CALL SURF
C
C CHECK FOR STACKED DATA
  IF (MORE .EQ. 1) GO TO 610
C
C CHECK IF A DUMMY STATUS 3 ELEMENT WANTED.
740 IF (IFLAP .EQ. 1) GO TO 10
  IF (ISTATS .NE. 1) CALL STATUS
C
C**END OF ALL CONFIGURATION COMPONENTS (GO TO STARTING STATEMENT)
  GO TO 10
C
750 WRITE (TAPEOT,760) I
760 FORMAT(1X, //,1H,31H** FATAL ERROR ** PROGRAM STOPPED.,
1 //SX,34HCONTROL FLAG CARD PARAMETER NUMBER,I3,4H IS ,
2 26HOUTSIDE THE ALLOWABLE RANGE.)
C
C**AFTER END OF FILE ON INPUT = STOP
770 STOP
C 780 RETURN
C 780 CONTINUE
C
  END

```

SURF

SURF 001
 SURF 002
 SURF 003
 SURF 004
 SURF 005
 SURF 006
 SURF 007
 SURF 008
 SURF 009
 SURF 010
 SURF 011
 SURF 012
 SURF 013
 SURF 014
 SURF 015
 SURF 016
 SURF 017
 SURF 018
 SURF 019
 SURF 020
 SURF 021
 SURF 022
 SURF 023
 SURF 024
 SURF 025
 SURF 026
 SURF 027
 SURF 028
 SURF 029
 SURF 030
 SURF 031
 SURF 032
 SURF 033
 SURF 034
 SURF 035

```

SURROUTINE SURF
C*****THIS SUBROUTINE DEFINES A GENERAL AIRFOIL SURFACE
C IN THE BODY-AXIS COORDINATE SYSTEM. THE SURFACE IS
C GENERATED ACCORDING TO THE NUMBER OF AIRFOIL SECTIONS, NAF,
C THE ORIENTATION OF THE CHORD LINE (YAW-PITCH-ROLL SEQUENCE),
C THE MEAN LINE DISTRIBUTION, AND THE THICKNESS DISTRIBUTION.
C THE SECTION COORDINATES MAY BE DEFINED AS AN NACA TYPE
C (THICKNESS DISTRIBUTION NORMAL TO MEAN LINE) OR AS A
C SHEARED TYPE (THICKNESS NORMAL TO THE CHORD LINE).
C
C AN OPTION TO CALCULATE THE TIP-PLANE SURFACE OR THE ROOT-
C PLANE SURFACE, OR BOTH, IS ALSO AVAILABLE.
C ALL INPUT QUANTITIES ENTER THROUGH THE LABELED COMMON/ARFOIL/.
C
C NAF - THE NUMBER OF AIRFOIL SECTIONS
C NAFORD - THE NUMBER OF SECTION ORDINATES
C NACA - = 1, CAMBER DATA INPUT
C XCF - = 1, NACA TYPE SECTION
C AFORG - ARRAY OF CHORD STATIONS, PERCENT CHORD
C AFCAM - ARRAY OF CHORD ORIGIN AND ORIENTATION DATA
C AFORD - ARRAY OF CAMBER DISTRIBUTION, PERCENT CHORD
C SURFID - ARRAY OF THICKNESS DISTRIBUTION, PERCENT CHORD
C ITTP - = 1, CALCULATE TIP SURFACE.
C IROOT - = 1, CALCULATE ROOT SURFACE.
C*****
C DIMENSION XOC(30),AFORG(20,7),AFCAM(20,30),AFORD(20,30),
C Y(2),Y(2),Z(2),COELTA(30),SDELTA(30)
C
C COMMON /ACF PINT,ITAPE,ITYPE,PI,DX,CASE
C
C COMMON /ARFOIL/ NAF,NAFORD,NCAM,NACA,ITIP,IROOT,
C XOC,AFORG,AFCAM,AFORD,SURFID
C ,ISURF,TELAP,NHNGE
C
C 1
C 2
  
```

SURF

SURF

COMMON /TAPE/TAPFIN,TAPEOT,TAPEA,TAPFB,TAPFC,TAPFD,TAPEE,TAPFF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPFK

C

INTEGER STAT1,STAT2

1 INTEGER TAPFIN,TAPEOT,TAPFA,TAPER,TAPFC,TAPFD,TAPEE,TAPFF,
TAPEG,TAPFH,TAPEI,TAPEJ,TAPEK

1 INTEGER CASE

C

DESCRIPTION OF COUNTERS AND CONTROLS

I IS THE AIRFOIL SECTION NUMBER

I = 1 IS THE ROOT

I = NAF IS THE TIP

J IS THE CHORDWISE STATION NUMBER

J = 1 IS THE LEADING EDGE

J = NAFORD IS THE TRAILING EDGE

K IS THE POINT COUNTER FOR TYPE 3 CARDS (TWO POINTS PER CARD).

IEND IS CONTROL FLAG FOR TYP OR ROOT SURFACE

INEXT IS SECTION SEQUENCE INCREMENT

INEXT = +1 FOR UPPER SURFACE

INEXT = -1 FOR LOWER SURFACE

ISFO IS COUNTER FOR OUTPUT DATA (CC'S 77-80)

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C*****THE SURFACE IS FIRST TESTED FOR POSSIBLE SYMMETRY

ABOUT THE X=Z ORIGIN PLANE. IF SO, THE SURFACE

SYMMETRY FLAG (ISYMM) IS SET TO +1 OR -1 AS PER SIGN OF PHI.

TO BE SYMMETRICAL THREE CONDITIONS MUST BE MET

1. NO CAMBER DISTRIBUTION

2. THE CHORD LINE IS IN THE X=Z ORIGIN PLANE

3. THE ROLL ANGLE PHI IS + OR - 90.0 DEGREES

THE TEST IS MADE ON BOTH THE ROOT AND TIP SECTIONS

C

I = 0

ISFO = 0

IEND = 0

SURF 036
SURF 037
SURF 038
SURF 039
SURF 040
SURF 041
SURF 042
SURF 043
SURF 044
SURF 045
SURF 046
SURF 047
SURF 048
SURF 049
SURF 050
SURF 051
SURF 052
SURF 053
SURF 054
SURF 055
SURF 056
SURF 057
SURF 058
SURF 059
SURF 060
SURF 061
SURF 062
SURF 063
SURF 064
SURF 065
SURF 066
SURF 067
SURF 068
SURF 069
SURF 070

SURF

SURF

SURF 071
 SURF 072
 SURF 073
 SURF 074
 SURF 075
 SURF 076
 SURF 077
 SURF 078
 SURF 079
 SURF 080
 SURF 081
 SURF 082
 SURF 083
 SURF 084
 SURF 085
 SURF 086
 SURF 087
 SURF 088
 SURF 089
 SURF 090
 SURF 091
 SURF 092
 SURF 093
 SURF 094
 SURF 095
 SURF 096
 SURF 097
 SURF 098
 SURF 099
 SURF 100
 SURF 101
 SURF 102
 SURF 103
 SURF 104
 SURF 105

```

ISYMM = 0
IHNCE = 1000
IF (ISURF .NE. 0) GO TO 230
IF (NCAM .EQ. 1) GO TO 10
IF (AFORG(1,2) .GT. 0.0001) GO TO 10
IF (AFORG(NAF,2) .GT. 0.0001) GO TO 10
IF (COS(AFORG(1,7)*PI/180.) .GT. 0.0001) GO TO 10
IF (COS(AFORG(NAF,7)*PI/180.) .GT. 0.0001) GO TO 10
C
C SURFACE IS SYMMETRICAL ABOUT X=Z ORIGIN PLANE.
ISYMM = 1
CHECK IF PHI IS POSITIVE OR NEGATIVE.
IF (AFORG(1,7) .LT. 0.0) GO TO 10
C
C POSITIVE VERTICAL, DO LOWER SURFACE FIRST.
ISYMM = 1
I = NAF + 1
GO TO 200
C
C
10 CONTINUE
INEXT = 1
PZ = 1.0
DETA = 1.0
C
20 K = 0
STAT1 = 2
STAT2 = 0
30 I = J + INEXT
JP = 1
IF (I .GT. NAF) GO TO 150
IF (I .EQ. 0) GO TO 150
C
C*****CALCULATE CHORD LENGTH C AND AIRFOIL ORIENTATION,
C SINF AND COSINE VALUES OF

```

SURF

SURF

SURF 106
 SURF 107
 SURF 108
 SURF 109
 SURF 110
 SURF 111
 SURF 112
 SURF 113
 SURF 114
 SURF 115
 SURF 116
 SURF 117
 SURF 118
 SURF 119
 SURF 120
 SURF 121
 SURF 122
 SURF 123
 SURF 124
 SURF 125
 SURF 126
 SURF 127
 SURF 128
 SURF 129
 SURF 130
 SURF 131
 SURF 132
 SURF 133
 SURF 134
 SURF 135
 SURF 136
 SURF 137
 SURF 138
 SURF 139
 SURF 140

```

C      YAW ANGLE = SPPI, CPSI
C      PITCH ANGLE = STHETA, CTHETA
C      ROLL ANGLE = SPHI, CPHI
C
      DXX = (AFORG(I,4) * AFORG(I,1)) * DX
      DYY = AFORG(I,2) * AFORG(I,5)
      DZZ = AFORG(I,6) * AFORG(I,3)
      DSI = DXX**2 + DYY**2
      DSP = SQRT(DSI + DZZ**2)
      DSI = SQRT(DSI)
C
      SPPI = DYY/DSI
      CPSI = DXX/DSI
C
      STHETA = DZZ/DS2
      CTHETA = DSI/DS2
C
      SPHI = SIN(AFORG(I,7) * PI/180.)
      CPHI = COS(AFORG(I,7) * PI/180.)
C
      C = 0.01 * DS2
C
C***  PREFSET MEAN LINE SLOPES
      DO 43 J = 1, NAFORD
      CDELTA(J) = 1.0
      SDDELTA(J) = 0.0
      40 SDDELTA(J) = 0.0
C
C***  CHECK IF CAMRER DATA INPUT
      IF (NCAM .NE. 1) GO TO 70
C
      IF (NACA .NE. 2) GO TO 46
      IF (C .EQ. 0.0) GO TO 46
      CI = 1.0/C
      DO 45 J = 1, NAFORD
      45 AFCAM(I,J) = AFCAM(I,J) * CI
  
```

SURF

SURF

```

141 SURF 141
142 SURF 142
143 SURF 143
144 SURF 144
145 SURF 145
146 SURF 146
147 SURF 147
148 SURF 148
149 SURF 149
150 SURF 150
151 SURF 151
152 SURF 152
153 SURF 153
154 SURF 154
155 SURF 155
156 SURF 156
157 SURF 157
158 SURF 158
159 SURF 159
160 SURF 160
161 SURF 161
162 SURF 162
163 SURF 163
164 SURF 164
165 SURF 165
166 SURF 166
167 SURF 167
168 SURF 168
169 CURF 169
170 SURF 170
171 SURF 171
172 SURF 172
173 SURF 173
174 SURF 174
175 SURF 175

```

```

46 CONTINUE
C
C*** CHECK AIRFOIL SECTION TYPE
IF (NACA .NE. !) GO TO 70
C
C*** NACA TYPE, THICKNESS NORMAL TO MEAN LINE.
MUST FIRST CALCULATE SLOPE ALONG MEAN LINE.
USE NEWTONIAN SINGLE QUADRATIC
C
J1 = 0
DO 60 IP = 1, NAFORD
C
IF (JP .EQ. 2 .OR. IP .EQ. NAFORD) GO TO 50
C
J1 = J1 + 1
J2 = J1 + 1
J3 = J1 + 2
X1 = XDC(J1)
X2 = XDC(J2)
X3 = XDC(J3)
Y1 = AFCAM(I,J1)
Y2 = AFCAM(I,J2)
Y3 = AFCAM(I,J3)
C
50 DYDX = (Y2-Y1)/(X2-X1) + (2.0*YDC(IP) - X2 - X1)*
1 ((Y3-Y2)/(X3-X2) - (Y2-Y1)/(X2-X1))/(X3-X1)
C
CDELTA(IP) = 1.0/SQRT(1.0 + DYDX**2)
SDELTA(IP) = DYDX*CDELTA(IP)
C
60 CONTINUE
C
C*** BEGIN SURFACE CALCULATION
70 J = 0

```

SURF

SURF

```

      IF (K .EQ. 1) GO TO 90
      80 K = 0
      90 J = J + 1
      IF (J .LF. NAFORD) GO TO 100
C*** SET DUMMY POINT AND WRITE DATA.
      K = 2
      X(K) = X(1)
      Y(K) = Y(1)
      Z(K) = Z(1)
      STAT2 = STAT1
      GO TO (110,260), JP
C
C*** CALCULATE COORDINATES
      100 K = K + 1
      XI = (XOC(J) - AFORD(I,J))*SDELTA(J)*DZ)*C
      ETA = (AFCAM(I,J) + DZ*AFORD(I,J))*CDELTA(J))*C
      X(K) = DX*AFORG(I,1) - XI*CTHETA*CPSI
      + DELTA*ETA*(STHETA*CPSI*CPI + SPSI*SPHI)
      Y(K) = AFORG(I,2) - XI*CTHETA*SPSI
      + DELTA*ETA*(STHETA*SPSI*CPI - CPSI*SPHI)
      Z(K) = AFORG(I,3) + XI*STHETA + DELTA*ETA*CTHETA*CPI
C
      IF (K .EQ. 1) GO TO 90
C
      GO TO (110,260), JP
C*** WRITE DATA
      110 ISEG = ISEG + 1
      WRITE(ITAPE,120) X(1),Y(1),Z(1),STAT1,
      X(2),Y(2),Z(2),STAT2,CASE,ITYPE,SURFID,ISEG
      I
C
      120 FORMAT(3F10.4,I1,3F10.4,I1,2X12,5X11,A4,I4)
C
      IF (JPRINT .EQ. 1)
      14WRITE (TAPEOT,130) X(1),Y(1),Z(1),STAT1,

```

SURF 176
 SURF 177
 SURF 178
 SURF 179
 SURF 180
 SURF 181
 SURF 182
 SURF 183
 SURF 184
 SURF 185
 SURF 186
 SURF 187
 SURF 188
 SURF 189
 SURF 190
 SURF 191
 SURF 192
 SURF 193
 SURF 194
 SURF 195
 SURF 196
 SURF 197
 SURF 198
 SURF 199
 SURF 200
 SURF 201
 SURF 202
 SURF 203
 SURF 204
 SURF 205
 SURF 206
 SURF 207
 SURF 208
 SURF 209
 SURF 210

SURF

SURF

```

      2      X(2),Y(2),Z(2),STAT2, CASE, ITYPE, SURFID, ISEQ
C 130 FORMAT(1X,3F10.4,I1,3F10.4,I1,2Y12,5X11,A4,I4)
C
C*** CHECK IF THIS IS TO HAVE A CONTROL SURFACE.
      IF (CJ.EQ. IMNGE) GO TO 250
C*** CHECK STATUS FLAGS
      IF (STAT1.GT. 0) STAT1 = 0
      IF (STAT2.GT. 0) STAT2 = 0
C
C*** CHECK FOR PROPER CYCLING
      IF (J .LT. NAFORD) GO TO 80
C
C
C*** HAVE COMPLETED AIRFOIL SECTION, SET STATUS AND GO TO NEXT ONE.
140 STAT1 = 1
      GO TO 30
C
C*** HAVE COMPLETED SURFACE, CHECK WHICH ONE.
150 CONTINUE
      IF (NACA.EQ. 2) NACA = 0
      IF (IFLAP.EQ. 1) GO TO 270
      IF (INEXT.LT. 0) GO TO 160
C
C*** JUST COMPLETED UPPER SURFACE.
      IF (IEND.EQ. 1) GO TO 190
      IF (ITIP.EQ. 1) GO TO 180
      IF (ISYMM.EQ. 0) GO TO 200
C
C*** JUST COMPLETED LOWER SURFACE.
160 IF (IEND = 1) 170,190,220
170 IF (ITIP.EQ. 1) GO TO 180
      IF (CROOT.EQ. 1) GO TO 210
      RETURN
C

```

SURF 211
 SURF 212
 SURF 213
 SURF 214
 SURF 215
 SURF 216
 SURF 217
 SURF 218
 SURF 219
 SURF 220
 SURF 221
 SURF 222
 SURF 223
 SURF 224
 SURF 225
 SURF 226
 SURF 227
 SURF 228
 SURF 229
 SURF 230
 SURF 231
 SURF 232
 SURF 233
 SURF 234
 SURF 235
 SURF 236
 SURF 237
 SURF 238
 SURF 239
 SURF 240
 SURF 241
 SURF 242
 SURF 243
 SURF 244
 SURF 245

SURF

SURF

SURF 246
 SURF 247
 SURF 248
 SURF 249
 SURF 250
 SURF 251
 SURF 252
 SURF 253
 SURF 254
 SURF 255
 SURF 256
 SURF 257
 SURF 258
 SURF 259
 SURF 260
 SURF 261
 SURF 262
 SURF 263
 SURF 264
 SURF 265
 SURF 266
 SURF 267
 SURF 268
 SURF 269
 SURF 270
 SURF 271
 SURF 272
 SURF 273
 SURF 274
 SURF 275
 SURF 276
 SURF 277
 SURF 278
 SURF 279
 SURF 280

SURF

```

C
C
C*** TIP SURFACE
180 STAT1 = 2
    IEND = 0
    INFXT = +1
    I = NAF
    IF (ISYMM .NE. 1) GO TO 70
    DELTA = 0.0
    I = Y - 1
    GO TO 30
190 DELTA = 1.0
    IF (ISYMM .EQ. -1) DELTA = 1.0
    IF (ISYMM .EQ. 0) DZ = -DZ
    ITIP = 0
    IEND = 0
    J = NAF
    J = 0
    GO TO 80
C
C
C*** LOWER SURFACE.
200 INFXT = -1
    DZ = -1.0
    DELTA = 1.0
    GO TO 20
C
C
C*** ROOT SURFACE
210 STAT1 = 2
    IEND = 2
    INFXT = -1
    I = 1
    DELTA = 1.0
  
```

SURF

SURF 281
SURF 282
SURF 283
SURF 284
SURF 285
SURF 286
SURF 287
SURF 288
SURF 289
SURF 290
SURF 291
SURF 292
SURF 293
SURF 294
SURF 295
SURF 296
SURF 297
SURF 298
SURF 299
SURF 300
SURF 301
SURF 302
SURF 303
SURF 304
SURF 305
SURF 306
SURF 307
SURF 308
SURF 309
SURF 310
SURF 311
SURF 312
SURF 313
SURF 314
SURF 315

```
IF (ISYMM .EQ. 0) GO TO 70  
I = 2  
IF (ISYMM .EQ. 1) GO TO 30  
DETA = 0.0  
GO TO 30  
220 DETA = 1.0  
IF (ISYMM .EQ. 1) DETA = 0.0  
IF (ISYMM .EQ. 0) DZ = -DZ  
IEND = 0  
I = 1  
J = 0  
GO TO 80
```

```
230 ISYMM = -ISURF  
C*** ONLY UPPER OR LOWER SURFACE, ZERO OL TIP AND ROOT SECTIONS.  
ITIP = 0  
IRMT = 0  
C  
C*** CHCK IF THIS SURFACE TO HAVE A FLAP (OR CONTROL SURFACE).  
IF (IFLAP .EQ. 0) GO TO 240  
C  
C*** CONTROL SURFACE. SET HINGE LINE LOCATION,  
IHNGE = NHNGE  
REWIND TAPEB  
240 IF (ISYMM .LT. 0) GO TO 10  
I = NAF + 1  
GO TO 200  
C  
C  
C
```

SURF

SURF

```
250 J = J - 1
    JP = 2
    STAT1 = 1
    GO TO 80
C
C
260 WRITE (TAPE8) X,Y,Z,STAT1,STAT2
    STAT1 = 0
    STAT2 = 0
    IF (J .LT. NAFORD) GO TO 80
C
C*** HAVE COMPLETED AIRFOIL SECTION
    STAT1 = 1
    GO TO 30
C
C*** HAVE COMPLETED AIRFOIL. TRANSFER FLAP ELEMENTS FROM
    UNIT 3 TO UNIT ITAPE. GIVE FIRST POINT STATUS 2
    AND LAST POINT STATUS 3.
C
270 CONTINUE
    REWIND TAPE8
    ITOTAL = NAF*(NAFORD + 1)/2
    READ (11) X,Y,Z,STAT1,STAT2
    STAT1 = 2
280 ISFO = ISEQ + 1
    IF (ISEQ .EQ. ITOTAL) STAT2 = 3
    WRITE (ITAPE,120) X(1),Y(1),Z(1),STAT1,
1      X(2),Y(2),Z(2),STAT2,CASE,I,TYPE,SURFID,ISEQ
C
    IF (IPRINT .EQ. 1)
2  IWRITE (TAPECT,130) X(1),Y(1),Z(1),STAT1,
    X(2),Y(2),Z(2),STAT2,CASE,I,TYPE,SURFID,ISEQ
C
    IF (ISEQ .EQ. ITOTAL) RETURN
    READ (TAPE8) X,Y,Z,STAT1,STAT2
```

SURF 316
SURF 317
SURF 318
SURF 319
SURF 320
SURF 321
SURF 322
SURF 323
SURF 324
SURF 325
SURF 326
SURF 327
SURF 328
SURF 329
SURF 330
SURF 331
SURF 332
SURF 333
SURF 334
SURF 335
SURF 336
SURF 337
SURF 338
SURF 339
SURF 340
SURF 341
SURF 342
SURF 343
SURF 344
SURF 345
SURF 346
SURF 347
SURF 348
SURF 349
SURF 350

SURF

SURF

SURF 351
SURF 352
SURF 353
SURF 354

GO TO 260

C
C

END

SURF

NACEL

```

C          SUPROUTINE  NACFL
          DIMENSION  PODORG(9,6),XPOD(9,30),PODR(9,30),X(2),Y(2),Z(2),
1          ICR(9),IEL(9),NFL(9)
C
C          COMMON /ACFT/PRINT,ITAPE,ITYPE,PI,DX,CASE
C
C          COMMON /NACE/D/NP,NPODR,PODRG,XPOD,PODR,IOR,IEL,NEL
C
C          COMMON /TAPE/TAPEIN,TAPEOT,TAPFA,TAPEB,TAPFC,TAPED,TAPEE,TAPFF,
1          TAPEG,TAPFH,TAPEI,TAPEJ,TAPEK
C          INTEGER  TAPFIN,TAPEOT,TAPEA,TAPER,TAPEC,TAPED,TAPEE,TAPFF,
1          TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C          INTEGER CASE
C          INTEGER STAT1,STAT2
C
C          I IS THE NACELLE OR POD NUMBR
C          J IS NUMBER IDENTIFYING AXIAL LOCATION
C          N IS THE POINT COUNTER ON THE SURFACE AT A GIVEN STATION
C
C          I = 0
C          I = I + 1
C          IF (I .GT. NP) GO TO 140
C          ISFO = 0
C
C          SET COUNTERS AND CONTROLS
C          STAT1 = 2
C          STAT2 = 0
C          J = 1
C
C          SET POD ORIGIN QUANTITIES
C          XO = PODORG(I,1)
C          YO = PODORG(I,2)
C          ZO = PODORG(I,3)

```

NACE 001
 NACE 002
 NACE 003
 NACE 004
 NACE 005
 NACE 006
 NACE 007
 NACE 008
 NACE 009
 NACE 010
 NACE 011
 NACE 012
 NACE 013
 NACE 014
 NACE 015
 NACE 016
 NACE 017
 NACE 018
 NACE 019
 NACE 020
 NACE 021
 NACE 022
 NACE 023
 NACE 024
 NACE 025
 NACE 026
 NACE 027
 NACE 028
 NACE 029
 NACE 030
 NACE 031
 NACE 032
 NACE 033
 NACE 034
 NACE 035

NACEL

NAGEL

NACE 036
 NACE 037
 NACE 038
 NACE 039
 NACE 040
 NACE 041
 NACE 042
 NACE 043
 NACE 044
 NACE 045
 NACE 046
 NACE 047
 NACE 048
 NACE 049
 NACE 050
 NACE 051
 NACE 052
 NACE 053
 NACE 054
 NACE 055
 NACE 056
 NACE 057
 NACE 058
 NACE 059
 NACE 060
 NACE 061
 NACE 062
 NACE 063
 NACE 064
 NACE 065
 NACE 066
 NACE 067
 NACE 068
 NACE 069
 NACE 070

NAGEL

```

C SET UP THE ROTATION ANGLES FOR POD ORIFNTANTION
  CTMETHA = 0.0
  CPSI = 1.0
  DXX = (XN - PDDORG(I,4))*DX
  DYY = YN - PDDORG(I,5)
  DZZ = -ZO + PDDORG(I,6)
  DS1 = DXX**2 + DYY**2
  DS2 = SQRT(DS1 + DZZ**2)
  IF (DS1 .LT. 0.0001) GO TO 20
  DS1 = SQRT(DS1)
  CPSI = DXX/DS1
  SPSI = DYY/DS1

C
  20 CTMETHA = DS1/DS2
  SYHETA = DZZ/DS2

C CALCULATE ELEMENT SPACING, DPHI.
  30 NR = 12
  IF (IEL(I) .NE. 1) GO TO 40
  IF (NEL(I) .GT. 0 .AND. NEL(I) .LE. 36) NR = NEL(I)
  40 DPHI = PI/NR

C CHECK POD SYMMETRY WITH RESPECT TO X-Z ORIGIN PLANE.
  IF (ABS(YO) .GT. 1.E-5) GO TO 50
  IF (IOR(I) .NE. 1) GO TO 60
  IF (ABS(PDDORG(I,5)) .LE. 1.E-5) GO TO 60
  50 NR = 2*NR
  60 NR = NR + 1

C NOTE. THE Y- AND Z-COORDINATES ARE DEFINED AS A FUNCTION

```

NACEL

```

C   OF ANGLE PHI WHICH IS DEFINED TO BE -PI/2 AT THE
C   NEGATIVE Z-AXIS. THIS DEFINITION ALLOWS THE SIGN
C   OF THE DELTA-Z INCREMENT TO BE AUTOMATICALLY ACCOUNTED FOR.
      TO N = 0
      PHI = -PI/2. -DPHI
C
      80 K = 0
      90 N = N + 1
      IF (N.LE. NR) GO TO 100
C SET DUMMY POINT.
      K = 2
      STAT2 = 0
      X(K) = X(1)
      Y(K) = Y(1)
      Z(K) = Z(1)
      GO TO 110
C
      100 K = K + 1
      PHI = PHI + DPHI
      YP = PDDR(I,J)*COS(PHI)
      ZP = PDDR(I,J)*SIN(PHI)
C
      X(K) = DX*XD + XPOD(I,J)*CTHETA*CPSI + YP*SPSI + ZP*STHETA*CPSI
C
      Y(K) = YD + XPOD(I,J)*CTHETA*SPSI + YP*CPSI + ZP*STHETA*SPSI
C
      Z(K) = ZD + XPOD(I,J)*STHETA + ZP*CTHETA
C
      IF (K.EQ. 1) GO TO 90
C
C WRITE DATA
      110 ISEQ = ISEQ + 1
      WRITE(ITAPE,120) X(1),Y(1),Z(1),STAT1,
      1      X(2),Y(2),Z(2),STAT2,CASF,ITYPE,I,ISEQ

```

NACE 071
 NACE 072
 NACE 073
 NACE 074
 NACE 075
 NACE 076
 NACE 077
 NACE 078
 NACE 079
 NACE 080
 NACE 081
 NACE 082
 NACE 083
 NACE 084
 NACE 085
 NACE 086
 NACE 087
 NACE 088
 NACE 089
 NACE 090
 NACE 091
 NACE 092
 NACE 093
 NACE 094
 NACE 095
 NACE 096
 NACE 097
 NACE 098
 NACE 099
 NACE 100
 NACE 101
 NACE 102
 NACE 103
 NACE 104
 NACE 105

NACEL

NACEL

NACE 106
NACE 107
NACE 108
NACE 109
NACE 110
NACE 111
NACE 112
NACE 113
NACE 114
NACE 115
NACE 116
NACE 117
NACE 118
NACE 119
NACE 120
NACE 121
NACE 122
NACE 123
NACE 124
NACE 125
NACE 126
NACE 127
NACE 128
NACE 129

```
C 120 FORMAT(3F10.4,I1,3F10.4,I1,2X12,5X11,#PDD#,I5,I4)
C
C      IF (IPRINT .EQ. 1)
C      1WRITE (TAPEQT,130) X(1),Y(1),Z(1),STAT1,
C      2      X(2),Y(2),Z(2),STAT2,CASE,I,ITYPE,I,YSFQ
C 130 FORMAT(1H,3F10.4,I1,3F10.4,I1,2X12,5X11,#PDD#,I1,I4)
C
C RESET STATUS FLAG
C      IF (STAT1 .GT. 0) STAT1 = 0
C
C CHECK FOR PROPER CYCLING
C      IF (N .LT. NR) GO TO 80
C
C HAVE COMPLETED AN AXIAL STATION. CHECK FOR MORE
C      J = J + 1
C      IF (J .GT. NPDDR) GO TO 10
C
C ONE OR MORE STATIONS LEFT. SET STATUS FLAG AND CONTINUE
C      STAT1 = 1
C      GO TO 70
C 140 RETURN
C      END
```

NACEL

FUSE

```

C SURROUTINE FUSE
C DIMENSION XFUS(4,30),NFORX(4),ZFUS(4,30),FUSARD(4,30),X(2),
      YFUSY(4,30,30),ZFUSZ(4,30,30),Y(2),Z(2),NRADX(4)
C COMMON /ACFT/PRINT,ITAPE,ITYPE,PI,DX,CASE
C COMMON /FUSD/J2,J6,NFUS,NRADX,NFORX,XFUS,ZFUS,YFUSY,ZFUSZ,FUSARD
C COMMON /TAPE/TAPEIN,TAPEOT,TAPFA,TAPEB,TAPEC,TAPEE,TAPEF,
      TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C INTEGER STAT1,STAT2
C INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
      TAPEG,TAPEH,TAPEI,TAPFJ,TAPEK
C INTEGER CASE
C I IS THE FUSELAGE SEGMENT NUMBER
C J IS THE AXIAL STATION NUMBER
C N IS THE SURFACE POINT NUMBER
C THERE ARE TWO MAIN PARTS TO THIS ROUTINE
C ONE SETS UP THE COORDS FOR AN ARBITRARY BODY
C THE OTHER IS FOR A BODY OF REVOLUTION GIVEN THE AREA DISTRIBUTION.
      K = 0
      I = 0
C TEST FOR BODY TYPE
      IF (J2 .NE. 1) GO TO 90
      IF (J6 .EQ. 1) GO TO 90
C ARBITRARY BODY
      10 I = I + 1
      IF (I .GT. NFUS) RETURN
      ISFO = 0
      STAT1 = 2

```

```

FUSE 001
FUSE 002
FUSE 003
FUSE 004
FUSE 005
FUSE 006
FUSE 007
FUSE 008
FUSE 009
FUSE 010
FUSE 011
FUSE 012
FUSE 013
FUSE 014
FUSE 015
FUSE 016
FUSE 017
FUSE 018
FUSE 019
FUSE 020
FUSE 021
FUSE 022
FUSE 023
FUSE 024
FUSE 025
FUSE 026
FUSE 027
FUSE 028
FUSE 029
FUSE 030
FUSE 031
FUSE 032
FUSE 033
FUSE 034
FUSE 035

```

FUSE

FUSE

FUSE 036
FUSE 037
FUSE 038
FUSE 039
FUSE 040
FUSE 041
FUSE 042
FUSE 043
FUSE 044
FUSE 045
FUSE 046
FUSE 047
FUSE 048
FUSE 049
FUSE 050
FUSE 051
FUSE 052
FUSE 053
FUSE 054
FUSE 055
FUSE 056
FUSE 057
FUSE 058
FUSE 059
FUSE 060
FUSE 061
FUSE 062
FUSE 063
FUSE 064
FUSE 065
FUSE 066
FUSE 067
FUSE 068
FUSE 069
FUSE 070

```
STAT2 = 0
J = 0
20 J = J + 1
   IF (J.GY. NFORX(I)) GO TO 10
   XO = YFUS(I,J)*DX
   N = 0
30 K = 0
40 N = N + 1
   IF (N.LE. NRADX(I)) GO TO 50
C SET UP DUMMY POINT.
  K = 2
  STAT1 = 0
  STAT2 = 0
  Y(K) = Y(I)
  Z(K) = Z(I)
  X(K) = X(I)
  GO TO 60
C
50 K = K + 1
  Y(K) = YFUSY(I,J,N)
  Z(K) = ZFUSZ(I,J,N)
  X(K) = XC
  IF (K.EQ. 1) GO TO 40
C
C WRITE DATA
60 ISFQ = ISEQ + 1
  WRITE(ITAPE,70) X(1),Y(1),Z(1),STAT1,
1             Y(2),Y(2),Z(2),STAT2 CASF,ITYPE,I,ISEQ
C
70 FORMAT(3F10.4,I1,3F10.4,I1,2XIP,5XII,1,1,I4)
C
  IF (IPRINT.EQ. 1)
  IWRITE (TAPEOT,80) X(1),Y(1),Z(1),STAT1;
```

FUSE

FUSE

```

      2      X(2),Y(2),Z(2),STAT2,CASE,ITYPE,I,ISEQ
C      80 FORMAT(1H,3F10.4,I1,3F10.4,I1,2X12,5X11,2FUS#,I1,I4)
C
C      IF (STAT1.NE.0) STAT1 = 0
C      IF (STAT2.NE.0) STAT2 = 0
C      CHECK FOR PROPER CYCLING
C      IF (N.LT.NRADX(I)) GO TO 30
C
C      HAVE COMPLETED AXIAL STATION. SET STATUS AND GO TO NEXT ONE.
C      STAT1 = 1
C      GO TO 20
C
C      BODY OF REVOLUTION
C      90 I = I + 1
C      IF (I.GT.NFUS) RETURN
C      STAT1 = 2
C      STAT2 = 0
C      ISFQ = 0
C
C      DPHI = PI/(NRADX(I)-1)
C      J = 0
C      100 J = J + 1
C      IF (J.GT.NFORX(I)) GO TO 90
C      X0 = XFUS(I,J)*DX
C      N = 0
C      PHI = -PI/2. + DPHI
C      R = SQRT(FUSARD(I,J)/PI)
C      110 K = 0
C      120 N = N + 1
C      IF (N.LE.NRADX(I)) GO TO 130
C
C      SET UP DUMMY POINT.

```

FUSE 071
FUSE 072
FUSE 073
FUSE 074
FUSE 075
FUSE 076
FUSE 077
FUSE 078
FUSE 079
FUSE 080
FUSE 081
FUSE 082
FUSE 083
FUSE 084
FUSE 085
FUSE 086
FUSE 087
FUSE 088
FUSE 089
FUSE 090
FUSE 091
FUSE 092
FUSE 093
FUSE 094
FUSE 095
FUSE 096
FUSE 097
FUSE 098
FUSE 099
FUSE 100
FUSE 101
FUSE 102
FUSE 103
FUSE 104
FUSE 105

FUSE

FUSE

FUSE 106
FUSE 107
FUSE 108
FUSE 109
FUSE 110
FUSE 111
FUSE 112
FUSE 113
FUSE 114
FUSE 115
FUSE 116
FUSE 117
FUSE 118
FUSE 119
FUSE 120
FUSE 121
FUSE 122
FUSE 123
FUSE 124
FUSE 125
FUSE 126
FUSE 127
FUSE 128
FUSE 129
FUSE 130
FUSE 131
FUSE 132
FUSE 133
FUSE 134
FUSE 135
FUSE 136
FUSE 137
FUSE 138
FUSE 139
FUSE 149

```

      K = 2
      STAT1 = 0
      STAT2 = 0
      Y(K) = Y(1)
      Z(K) = Z(1)
      X(K) = X(1)
      GO TO 140

C     130 K = K + 1
      X(K) = XD
      PHI = PHI + DPHI
      Y(K) = R*COS(PHI)
      Z(K) = ZFUS(I,J) + R*SIN(PHI)
      IF (K .EQ. 1) GO TO 120

C
C     WRITE DATA
C     140 ISEQ = ISEQ + 1
      WRITE(ITAPE,70) X(1),Y(1),Z(1),STAT1,
1      X(2),Y(2),Z(2),STAT2,CARF,ITYPE,I,ISEQ

C
C     IF (IPRINT .EQ. 1)
1WRITE (TAPEOT,60) X(1),Y(1),Z(1),STAT1,
2      X(2),Y(2),Z(2),STAT2,CARF,ITYPE,I,ISEQ

C
C     IF (STAT1 .GT. 0) STAT1 = 0
      IF (STAT2 .GT. 0) STAT2 = 0

C CHECK FOR PROPER CYCLING
      IF (N .LT. NRADX(I)) GO TO 110

C HAVE COMPLETED AXIAL STATION. SET STATUS AND CONTINUE.
      STAT1 = 1
      GO TO 100
```

FUSE

FUSE

FUSE 141
FUSE 142

FUSE

C
END

STATS3

STAT 001
 STAT 002
 STAT 003
 STAT 004
 STAT 005
 STAT 006
 STAT 007
 STAT 008
 STA - 009
 STAT 010
 STAT 011
 STAT 012
 STAT 013
 STAT 014
 STAT 015
 STAT 016
 STAT 017
 STAT 018
 STAT 019
 STAT 020
 STAT 021
 STAT 022
 STAT 023
 STAT 024
 STAT 025
 STAT 026
 STAT 027
 STAT 028

```

SURROUTINE STATS3
C THIS ROUTINE PREPARES A DUMMY; ZERO-AREA ELEMENT TO PROVIDE THE
C FINAL STATUS FLAG OF 3
C
C
COMMON /ACFT/IPRINT,ITAPE,ITYPF,PI
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEF,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEF,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
WRITE (ITAPE,10)
10 FORMAT (51H 0.0 0.0 0.0 0.0 2 0.0 0.0 0.0 )
1 29H 0.00010 3 )
WRITE (ITAPE,20)
20 FORMAT (51H -0.0001 0.0 0.0 1 -0.0001 0.0 )
1 29H 0.00013 )
IF (IPRINT .EQ. 1) WRITE (TAPEOT,30)
IF (IPRINT .EQ. 1) WRITE (TAPEOT,40)
30 FORMAT(52H 0.0 0.0 0.0 2 0.0 0.0 )
1 29H 0.00010 3 )
40 FORMAT(52H -0.0001 0.0 0.0 1 -0.0001 0.0 )
1 29H 0.00013 3 )
C RETURN
C FND

```

STATS3

AFRO

OVERLAY (MARK4,2,0)
PROGRAM AERO
SUBROUTINE AERO

C
C
C
C
C

*** THIS ROUTINE CONTROLS THE AERODYNAMIC PROGRAM

COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,PPEND,PFS,TPS,AFS,RHDFS,VIS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /FSBS/FS,RS
COMMON /TAGS/ITAG4,ITAG9,ITAG10
COMMON /TAPE/TAPEIN,TAPECT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
1 COMMON /REF/SRFF,MAC,SPAN,XCG,YCG,ZCG
COMMON /ABDATA/NAB,ALPHA(20),BETA(20),PDL(20),CDELTA(20),QI(20),
RI(20),PI(20)
1 COMMON /INTERF/INT,ISHE(20),NSHE(100),I9,ISHFF,INF(20,22),
DINF(20,6),LTOTAB(20)
1 DIMENSION TITLE(15),IPG(20),FS(8),BS(8)
INTEGER ERROR,PAGE
INTEGER TAPEIN,TAPECT,TAPEA,TAPEB,TAPEE,TAPEF,TAPEG,TAPEH,TAPEI,
TAPEJ,TAPEK
1 REAL MACH,MAC

C

DATA GAS1,GAS2,GAS3,GAS4/4M *A,4MIR* :4M+HEL,4MIUM*/

C
C

READ AERO SYSTEM CONTROL CARD
READ (TAPEIN,10) TITLE
10 FORMAT (15A0)
WRITE (TAPECT,20) TITLE
20 FORMAT (I1,31M**** AERODYNAMIC PROGRAM *****/1M0,15A0)

C
C

READ AERO FLAG CARD
READ (TAPEIN,30) IPG

AERO 001C
AERO 002C
AERO 003I
AERO 004
AERO 005
AERO 006
AERO 007
AERO 008
AERO 009
AERO 010
AERO 011
AERO 012
AERO 013
AERO 014
AERO 015
AERO 016
AERO 017
AERO 018
AERO 019
AERO 020
AERO 021
AERO 022
AERO 023
AERO 024
AERO 025
AERO 026
AERO 027
AERO 028
AERO 029
AERO 030
AERO 031
AERO 032
AERO 033
AERO 034
AERO 035

AERO

AERO

AERO 036
 AERO 037
 AERO 038
 AERO 039
 AERO 040
 AERO 041
 AERO 042
 AERO 043
 AERO 044
 AERO 045
 AERO 046
 AERO 047
 AERO 048
 AERO 049
 AERO 050
 AERO 051
 AERO 052
 AERO 053
 AERO 054
 AERO 055
 AERO 056
 AERO 057
 AERO 058
 AERO 059
 AERO 060
 AERO 061
 AERO 062
 AERO 063
 AERO 064
 AERO 065
 AERO 066
 AERO 067
 AERO 068
 AERO 069
 AERO 070

```

C
30 FORMAT (20I1)
   00 80 I=1,20
   IF (IPG(I) .EQ. 0) GO TO 90
   IF (IPG(I) .EQ. 1) WRITE (TAPE0,40) I
40 FORMAT (1H0,15X,12,31H FLOW FIELD ANALYSIS (OPTION 1))
   IF (IPG(I) .EQ. 2) WRITE (TAPE0,50) I
50 FORMAT (1H0,15X,12,31H SHIELDING ANALYSIS (OPTION 2) )
   IF (IPG(I) .EQ. 3) WRITE (TAPE0,60) I
60 FORMAT (1H0,15X,12,31H INVISCID PRESSURES (OPTION 3) )
   IF (IPG(I) .EQ. 4) WRITE (TAPE0,70) I
70 FORMAT (1H0,15X,12,31H VISCOUS FORCES (OPTION 4) )
   IF (IPG(I) .EQ. 5) WRITE (TAPE0,71) I
71 FORMAT (1H0,15X,12,31H SPECIAL ROUTINES (OPTION 5) )
80 CONTINUE

C
C READ FLIGHT CONDITION CARD
90 READ (TAPEIN,100) MACH,ALT,PSTAG,TSTAG,IGAS,NAB
100 FORMAT (4F10,0,I1,I2)
C READ REFERENCE DIMENSION CARD
   READ (TAPEIN,110) SRFF,MAC,SPAN,XCG,YCG,ZCG
110 FORMAT (6F10,0)
   DO 130 I=1,NAB
   READ (TAPEIN,120) ALPHA(I),BETA(I),RCL(I),CDELTA(I),G(I),PI(I),
   I
120 FORMAT (7F10,0)
130 CONTINUE

C
C SET GAS CONSTANTS
   IF (IGAS .EQ. 1) GO TO 140
C***AIR****
   GAM = 1.40
   GASCP = 6.00793E+3
   PRAN = 0.71
   AV1 = 225.0
  
```

AERO

AERO

AERO 071
 AERO 072
 AERO 073
 AERO 074
 AERO 075
 AERO 076
 AERO 077
 AERO 078
 AERO 079
 AERO 080
 AERO 081
 AERO 082
 AERO 083
 AERO 084
 AERO 085
 AERO 086
 AERO 087
 AERO 088
 AERO 089
 AERO 090
 AERO 091
 AERO 092
 AERO 093
 AERO 094
 AERO 095
 AERO 096
 AERO 097
 AERO 098
 AERO 099
 AERO 100
 AERO 101
 AERO 102
 AERO 103
 AERO 104
 AERO 105

AERO

```

AV2 = 8.0382434E+10
AV3 = 1.0
GTYPE(1) = GAS1
GTYPE(2) = GAS2
GO TO 150

C***HELIUM***
140 GAM = 5./3.
GASCP = 3.1250E+4
PRAN = 2./3.
AV1 = 1.0E+5
AV2 = 7.17884E+9
AV3 = 0.647
GTYPE(1) = GAS3
GTYPE(2) = GAS4

C
150 CONTINUE
C DETERMINE FREE STREAM PROPERTIES
IF (ALT .GE. -1000.0) GO TO 160
PFS = PSTAG
TFS = TSTAG
GO TO 180

160 CONTINUE
IF (PSTAG .GT. 0.0) GO TO 170
USE U S 1962 ATMOSPHERE
CALL ATMOS (ALT,TFS,PFS,AFS,RHQFS)
GO TO 190

C USE WIND TUNNEL CONDITIONS (GAMMA=GAM) F0=00,43,29C,26 OF TR=1135
170 PFS=PSTAG*(1.+(GAM-1.)*MACH**MACH/2.)**(-GAM/(GAM-1.))*2116.217
TFS = (TSTAG + 459.6) / (1.0 + (GAM-1.0)*MACH**MACH/2.0)

180 CONTINUE
AFS = SORT((GAM-1.)*GASCP*TFS)
RHQFS = GAM*PFS/((GAM-1.0)*GASCP*TFS)
190 IF(TFS.GE.AV1) VIS = 2.27E-8*TFS**1.5/PFS + 198.6)
IF(TFS.LT.AV1) VIS = AV2*TFS**AV3
  
```

AERO

AERO 106
AERO 107
AERO 108
AERO 109
AERO 110
AERO 111
AERO 112
AERO 113
AERO 114
AERO 115
AERO 116
AERO 117
AERO 118
AERO 119
AERO 120
AERO 121
AERO 122
AERO 123
AERO 124
AERO 125I
AERO 126C
AERO 127
AERO 128I
AERO 129C
AERO 130
AERO 131I
AERO 132C
AERO 133
AERO 134I
AERO 135C
AERO 136
AERO 137I
AERO 138C
AERO 139
AERO 140

AERO

```
V = MACH * AFS  
PENC = RHOFS * V / VIS  
C INITIALIZE FREE STREAM DATA ARRAY  
FS(1) = RHOFS  
FS(2) = PFS  
FS(3) = TFS  
FS(4) = AFS  
FS(5) = VIS  
FS(6) = MACH  
FS(7) = MACH * FS(4)  
FS(8) = FS(1) * FS(7) / FS(5)  
  
C  
J = 0  
200 J = J + 1  
IF (IPG(J), EQ, 0) GO TO 260  
IPRG = IPG(J)  
  
C  
GO TO (210,220,230,240,241,243), IPRG  
  
C  
210 CALL FLOW  
210 CALL OVERLAY (5HMARK0,2,1,6HRECALL)  
GO TO 250  
C  
220 CALL SHIELD  
220 CALL OVERLAY (5HMARK0,2,2,6HRECALL)  
GO TO 250  
C  
230 CALL PRES  
230 CALL OVERLAY (5HMARK0,2,3,6HRECALL)  
GO TO 250  
C  
240 CALL VISCUS  
240 CALL OVERLAY (5HMARK0,2,4,6HRECALL)  
GO TO 250  
C  
241 CALL SPEC  
241 CALL OVERLAY (5HMARK0,2,5,6HRECALL)  
GO TO 250  
C
```

AERO

```
C 243 CALL STREAM  
243 CALL OVERLAY (5HMARK0,2,6,6HRECALL)  
GO TO 250  
C 250 IF (ERROR .EQ. 0) GO TO 200  
GO TO 280  
260 CONTINUE  
C WRITE (TAPE07,270)  
270 FORMAT (1H0,37H *** AERODYNAMIC ANALYSIS IS COMPLETE )  
C  
C 280 RETURN  
280 CONTINUE  
END
```

AERO 141J
AERO 142C
AERO 143
AERO 144
AERO 145
AERO 146
AERO 147
AERO 148
AERO 149
AERO 150
AERO 151
AERO 152
AERO 153I
AERO 154C
AERO 155

AERO

COMPR

```

C SURROUTINE COMPR (ANGLE, MER, IPRINT, CPSYAG, ISDET, IFIRST, CP)
C USING THE FREE STREAM MACH NUMBER AND THE EQUIVALENT WEDGE ANGLE,
C THIS ROUTINE COMPUTES THE CONDITIONS BEHIND THE SHOCK
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FSBS/FS, RS
COMMON /GASD/GAM, GASCP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
COMMON /TAPE/TAPEIN, TAPEY, TAPEA, TAPEE, TAPEF, TAPEFC, TAPEJ, TAPEK,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
DIMENSION FS(8), ANGLE(3), RS(8), R(3)
DIMENSION YTITLE(15)
C
INTEGER CASF, PAGE, ERROR
INTEGER TAPFIN, TAPECT, TAPFA, TAPEB, TAPEF, TAPEFC, TAPED, TAPEE, TAPEF,
1 TAPFG, TAPEH, TAPEI, TAPEJ, TAPEK
REAL MACHSO, MACHD, MACH1
MER = 0
G = GAM
IF (ISDET .EQ. 1) GO TO 210
IF (ABS(ANGLE(2)) .LT. 0.00001) GO TO 240
IF (ANGLE(2) .GT. 55.0) GO TO 10
IF (ABS(ANGLE(2)) .LE. 2.0) GO TO 260
C
C SET UP CURIC TO BE SOLVED FOR SIN**2 THETA (SHOCK ANGLE) = CONSTANTS
C DEFINED IN EQUATION 150R OF TR 1135
B = -(FS(6)**2+2.)/FS(6)**2 -G*SIN(ANGLE(2))/57.29578)**2
C = (2.0*FS(6)**2 + 1.0)/FS(6)**4 + ((G+1.))**2/4. + (G-1.)/FS(6)**2)**2
1 SIN(ANGLE(2))/57.29578)**2
D = -COS(ANGLE(2))/57.29578)**2 /FS(6)**4
C
C CHECK FOR SHOCK DETACHMENT
IF ( (-B**2/9. + C/3.))**3 + ((8/3.))**3 = (8*C + 3.*D)/6.))**2)
1 .GE. 0.0) GO TO 10

```

COMP 001
 COMP 002
 COMP 003
 COMP 004
 COMP 005
 COMP 006
 COMP 007
 COMP 008
 COMP 009
 COMP 010
 COMP 011
 COMP 012
 COMP 013
 COMP 014
 COMP 015
 COMP 016
 COMP 017
 COMP 018
 COMP 019
 COMP 020
 COMP 021
 COMP 022
 COMP 023
 COMP 024
 COMP 025
 COMP 026
 COMP 027
 COMP 028
 COMP 029
 COMP 030
 COMP 031
 COMP 032
 COMP 033
 COMP 034
 COMP 035

COMPR

COMPR

COMP 036
 COMP 037
 COMP 038
 COMP 039
 COMP 040
 COMP 041
 COMP 042
 COMP 043
 COMP 044
 COMP 045
 COMP 046
 COMP 047
 COMP 048
 COMP 049
 COMP 050
 COMP 051
 COMP 052
 COMP 053
 COMP 054
 COMP 055
 COMP 056
 COMP 057
 COMP 058
 COMP 059
 COMP 060
 COMP 061
 COMP 062
 COMP 063
 COMP 064
 COMP 065
 COMP 066
 COMP 067
 COMP 068
 COMP 069
 COMP 070

C SHOCK NOT DETACHED, COMPUTE THREE REAL ROOTS

```

Y = B**2 * 3.*C
Z = (9.*B*C - 2.*B**3 - 27.*D)/(2.*Y**1.5)
W = ARCCS(Z)
Z = W/3.
Y = 2.*SQRT(Y)
R(1) = (Y*COS(Z) - B)/3.
R(2) = -(Y*COS(Z) + 60./57.29578) + R)/3.
R(3) = -(Y*COS(Z) - 60./57.29578) + R)/3.
GO TO 20
  
```

C CUBIC SOLUTION WAS NOT FOUND BECAUSE THE SHOCK HAS DETACHED. FLOW
 C PROPERTIES WILL BE CALCULATED BY THE METHOD OF KAUFMAN

```

10 ETAC = 1.0
TSF = 0
CALL NEWTPM (ANGLE,EMN,CP,ETAC,IPRINT,MER,CPSTAG,
1 ISE,IFIRST)
IF (MER - 1)230,210,210
  
```

C A SOLUTION TO THE CUBIC WAS FOUND. CHECK FOR DESIRED SOLUTION.
 C SMALLEST ROOT REQUIRES A DECREASE IN ENTROPY WHICH IS NOT ALLOWED,
 C LARGEST ROOT IS NOT ATTAINED IN PRACTICAL CASES. THEREFORE PICK
 C MIDDLE ANSWER.

```

20 IF (R(1) - W(2))30,110,40
30 K = 1
GO TO 50
40 K = 2
50 IF (R(2) - R(3))60,120,70
60 M = 1
GO TO 80
70 M = 2
80 IF (K.EQ. M) GO TO 120
IF (R(1) - R(3))90,130,100
90 GO TO (130,110), K
  
```

COMPR

COMPR

COMP 071
 COMP 072
 COMP 073
 COMP 074
 COMP 075
 COMP 076
 COMP 077
 COMP 078
 COMP 079
 COMP 080
 COMP 081
 COMP 082
 COMP 083
 COMP 084
 COMP 085
 COMP 086
 COMP 087
 COMP 088
 COMP 089
 COMP 090
 COMP 091
 COMP 092
 COMP 093
 COMP 094
 COMP 095
 COMP 096
 COMP 097
 COMP 098
 COMP 099
 COMP 100
 COMP 101
 COMP 102
 COMP 103
 COMP 104
 COMP 105

```

100 GO TO (110,130), K
110 ANGLE(3) = R(1)
    GO TO 140
120 ANGLE(3) = R(2)
    GO TO 140
130 ANGLE(3) = R(3)

C CHECK IF ANGLE IS NEGATIVE AND PRINT ERROR NOTE IF REQUIRED
140 IF (ANGLE(3) .GE. 0.0) GO TO 170
    IF (IPRINT .NE. 1) GO TO 160
    WRITE (TAPE07,150)
150 FORMAT (1H0,39H NEGATIVE VALUE FOUND FOR SIN**2 THETA
    1 41PHN CUBIC. TO CONTINUE, IT IS SET TO ZERO. )
160 ANGLE(3) = 0.0
170 IF (ANGLE(3) .LE. 1.0) GO TO 200
    IF (IPRINT .NE. 1) GO TO 190
    WRITE (TAPE07,180)
180 FORMAT (1H0,41H IN CUBIC, SIN**2 THETA GREATER THAN ONE.
    1 31H TO CONTINUE, IT IS SET TO ONE. )
190 ANGLE(3) = 1.0

C CALCULATE CONDITIONS BEHIND THE SHOCK USING THE SELECTED SIN**2 THETA
200 EMN = FS(6)**2 * ANGLE(3)
    DFNSITY EQ, 129 OF TR 1135
210 IF (EMN .LT. 1.01) EMN = 1.01
    PS(1) = FS(1)*(G+1.)*EMN/(G+1.)*EMN + 2)
C PRESSURE EQ, 128
    BS(2) = FS(2)*(2.*G*FMM + (G+1.))/(G+1.)
C TEMPERATURE EQ, 130
    R(3) = (2.*G*FMM -(G+1.))*(2.+(G+1.)*FMM)/((G+1.))**2*EMN)
    BS(3) = FS(3) * R(3)
C SPEED OF SOUND
    BS(4) = FS(4) * SQRT(R(3))
C VISCOSITY
    IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.6)
  
```

COMPR

COMPR

```

IF(BS(3).LT.AV1) BS(5) = AV2*BS(3)**AV1
C MACH NUMBER EG, .32
BS6SQ = ((G+1.)*FS(6))**2*EMN = 4.*(EMN-1.)*(G+EMN+1.))/
1 ((2.*G*EMN + (G-1.))*2. + (G-1.)*EMN)
IF (BS6SQ .LT. 1.0) BS(6) = 1.01
IF (BS6SQ .GE. 1.0) BS(6) = SQRT(BS6SQ)
C VELOCITY
BS(7) = BS(4) * BS(6)
C REYNOLDS NUMBER PER FOOT
BS(8) = RS(1) * BS(7)/RS(5)
MEP = 0
C SHOCK ANGLE
ANGLE(3) = SQRT(ANGLE(3))
IF (ABS(ANGLE(3)) .GT. 1.0) ANGLE(3) = 1.0
ANGLE(3) = ARSIN(ANGLE(3)) * 0.57295781E02
C CP = 4.*(EMN = 1.0)/((G+1.)*FS(6)**2)
C 230 RETURN
C ANGLE(2) IS ZERO, SET BS(I) = FS(I), CP = 0.0, AND EXIT

240 DO 250 I= 1,N
250 BS(I) = FS(I)
CP = 0.0
RETURN
C USE WEAK OBLIQUE SHOCK RELATIONSHIP (LIEPMAN AND RDSHKO, P.92)
260 ANGLE(3) = 1.0/FS(6)**2 + 0.5*(G+1.)/SQRT(FS(6)**2-1.0)*ANGLE(2)/
1 57.295779
GO TO 200
END

```

COMP 106
 COMP 107
 COMP 108
 COMP 109
 COMP 110
 COMP 111
 COMP 112
 COMP 113
 COMP 114
 COMP 115
 COMP 116
 COMP 117
 COMP 118
 COMP 119
 COMP 120
 COMP 121
 COMP 122
 COMP 123
 COMP 124
 COMP 125
 COMP 126
 COMP 127
 COMP 128
 COMP 129
 COMP 130
 COMP 131
 COMP 132
 COMP 133
 COMP 134
 COMP 135
 COMP 136

COMPR

EXPAND

```

C      SUBROUTINE EXPAND (ANGLE,MFR,IPRINT,ISDET,CF)
C
C      GIVEN THE FREE STREAM CONDITIONS (FS) AND THE TURNING ANGLE IN
C      DEGREES (ANGLE(2)), THIS SUBROUTINE PERFORMS AN ISENTROPIC PRANDTL-
C      MEYER EXPANSION(ANGLE(2),GT,0.) OR COMPRESSION(ANGLE(2),LT,0.)
C
C      COMMON /TAPE/TAPFIN,TAPED,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
C      TAPEG,TAPEH,TAPEI,TAPEJ,TAPFK
C      COMMON /EXEC/CASE,TITLE,PAGE,ERROR
C      COMMON /FSBS/FS,BS
C      COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
C      DIMENSION FS(8),ANGLE(3),RS(8),A(2),C(2)
C      DIMENSION TITLE(15)
C
C      INTEGER CASF,PAGE,ERROR
C      INTEGER TAPFIN,TAPED,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
C      TAPFG,TAPEH,TAPEI,TAPEJ,TAPEK
C      REAL NU1, NU2, NU1D, NU2D
C
C      CHECK IF FREE STREAM MACH NO. .GE. 1.0
C      IF (FS(6)).GE.1.0) GO TO 10
C      INPUT MACH NO. SUBSONIC. FOR PROGRAM CONTINUITY SET = 1.0 AND GO ON
C      EMSO = 1.0
C      GO TO 20
C
C      SQUARE FREE STREAM MACH NO.
C      10 EMSO = FS(6)**2
C      DEFINE GAMMA RATIO FUNCTION GR.
C      20 G = GAM
C      GR = SQRT((G+1.)/(G-1.))
C      CALCULATE PRANDTL-MEYER ANGLE FOR FREE STREAM CONDITIONS USING
C      EQUATION 17IC OF TR 1155 (RADIAN)
C      NU1 = GR*ATAN(SORT (EMSQ-1.)/GR ) = ATAN(SORT(EMSQ-1.))
C
C      CALCULATE PRANDTL-MEYER ANGLE AFTER THE EXPANSION (RADIAN)

```

EXPA 001
EXPA 002
EXPA 003
EXPA 004
EXPA 005
EXPA 006
EXPA 007
EXPA 008
EXPA 009
EXPA 010
EXPA 011
EXPA 012
EXPA 013
EXPA 014
EXPA 015
EXPA 016
EXPA 017
EXPA 018
EXPA 019
EXPA 020
EXPA 021
EXPA 022
EXPA 023
EXPA 024
EXPA 025
EXPA 026
EXPA 027
EXPA 028
EXPA 029
EXPA 030
EXPA 031
EXPA 032
EXPA 033
EXPA 034
EXPA 035

EXPAND

EXPAND

```

C      NU10 = NU1 * 57.295779
C      NU20 = NU10 + ANGLE(2)
C      NU2 = NU20/57.295779
C
C      CHECK IF FLOW COMPRESSED TO SUBSONIC.
C
C      IF (NU2D.GT.0.) GO TO 30
C      NU2D = LF. 0.0, RETURN SONIC CONDITIONS
C
C      RS(6) = 1.0
C      MER = 2
C      GO TO 190
C
C      CHECK IF FLOW HAS EXPANDED TO AN INFINITE MACH NUMBER (TAKEN
C      AS 100. FOR ALL PRACTICAL PURPOSES). IF $0, RETURN ZERO PRESSURE.
C      30 IF (NU2D.LT.(0.97815*(GR-1.)*99.0)) GO TO 40
C      MER = 2
C      RS(6) = 100.0
C      GO TO 190
C
C      START OF ITERATION TO FIND MACH NO. DOWNSTREAM
C      SFT INITIAL CONDITIONS AND TOLERANCE
C      40 I = 0
C      A(2) = 0.0
C      C(2) = 0.0
C      EPS = 1.E-4
C      JPATH = 1
C      JPATH CONTROLS THE LOGICAL PATH DURING THE ITERATION CYCLE.
C      CALCULATE APPROXIMATE DOWNSTREAM MACH NO.
C      RS(6) = FS(6)*(1. + (NU2-NU1)*(1.+5*(G-1.)*FMSG)/SQRT(EMSQ-1.))
C      IF (RS(6).GT.1.0) GO TO 50
C      RS(6) = 1.01
C      SFT ITERATION COUNTER AND CHECK FOR MAXIMUM
C      50 I = I + 1

```

EXPA 036
EXPA 037
EXPA 038
EXPA 039
EXPA 040
EXPA 041
EXPA 042
EXPA 043
EXPA 044
EXPA 045
EXPA 046
EXPA 047
EXPA 048
EXPA 049
EXPA 050
EXPA 051
EXPA 052
EXPA 053
EXPA 054
EXPA 055
EXPA 056
EXPA 057
EXPA 058
EXPA 059
EXPA 060
EXPA 061
EXPA 062
EXPA 063
EXPA 064
EXPA 065
EXPA 066
EXPA 067
EXPA 068
EXPA 069
EXPA 070

EXPAND

EXPAND

EXPA 071
 EXPA 072
 EXPA 073
 EXPA 074
 EXPA 075
 EXPA 076
 EXPA 077
 EXPA 078
 EXPA 079
 EXPA 080
 EXPA 081
 EXPA 082
 EXPA 083
 EXPA 084
 EXPA 085
 EXPA 086
 EXPA 087
 EXPA 088
 EXPA 089
 EXPA 090
 EXPA 091
 EXPA 092
 EXPA 093
 EXPA 094
 EXPA 095
 EXPA 096
 EXPA 097
 EXPA 098
 EXPA 099
 EXPA 100
 EXPA 101
 EXPA 102
 EXPA 103
 EXPA 104
 EXPA 105

```

C CHECK NUMBER OF ITERATIONS COUNTER
60 IF (I .LE. 20) GO TO 80
  WRITE (TAPE07,70) I
70 FORMAT (1H0,I4,42H ITERATIONS IN EXPANSION ROUTINE. THE LAST
  1 25H VALUE HAS BEEN ACCEPTED. )
  GO TO 190
C
80 A(2) = BS(6)
  R = (NU2 + ATAN(SORT(BS(6)**2 - (.0))) / GR
  R = TAN(R)
  BS(6) = SORT(1.0 + (R*GR)**2)
  C(2) = BS(6)
C
C CHECK IF FLOW ITERATION IS TO BE PRINTED OUT
IF (IPRINT .NE. 1) GO TO 100
WRITE (TAPE07,90) A(2),C(2)
90 FORMAT (1H ,17X3HMA#F6.4,4X3HMC#F8.4 )
C
C CHECK IF ITERATION ACCURACY HAS BEEN REACHED
100 DCA2 = C(2)-A(2)
  IF (ABS(DCA2/C(2)) .LE. EPS) GO TO 190
C
  GO TO (110,150,160), JPATH
110 JPATH = 2
C STEP ASSUMED VALUE BY AN ARBITRARY INCREMENT
C EXPERIENCEF HAS SHOWN THAT ONE-12TH OF C(2) TO BE A GOOD VALUE.
120 DA = C(2)/12.
130 A(1) = A(2)
  C(1) = C(2)
  DCA1 = DCA2
  IF (DCA1.GT.0.0) GO TO 140
  BS(6) = C(1) - DA
  
```

EXPAND

EXPAND

```

C MAKE SURE THAT 2ND GUESS IS NOT OUT OF RANGE.
  IF(BS(6).GT.1.0) GO TO 50
  BS(6) = (C(1)-1.0)/2. + 1.0
  GO TO 50
140 BS(6) = C(1) + DA
  GO TO 50
150 IF ((DCA2/DCA1).GT.0.0) GO TO 120
  JPATH = 3
160 IF ((DCA2/DCA1).GT.0.0) GO TO 150
C CALCULATE MACH NUMBER AFTER EXPANSION USING 2 PREVIOUS ESTIMATES
  IF ((C(2)=C(1)) .NE. 0.0) GO TO 170
  DADC = 0.0
  GO TO 180
170 DADC = (A(2)-A(1))/(C(2)=C(1))
180 BS(6) = (A(1)=C(1)*DADC)/(1.=DADC)
  A(1) = A(2)
  C(1) = C(2)
  DCA1 = DCA2
  DA = DA/2.
  GO TO 50

C CALCULATE FINAL CHARACTERISTICS BEHIND EXPANSION FAN
190 ONEOM = 1.0 / BS(6)
  ANGLF(3) = ARSIN(ONEOM)*57.295779
  Z = (2.0 + (G-1.)*EMSQ) / (2.0 + (G-1.)*BS(6)**2)
  BS(1) = FS(1) *Z**(1./(G-1.))
  BS(2) = FS(2) *Z**(G/(G-1.))
  BS(3) = FS(3) *Z
  BS(4) = FS(4) *SQRT(Z)
  IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.6)
  IF(BS(3).LT.AV1) BS(5) = AV2*BS(3)**AV3
  BS(7) = BS(6) * BS(4)
  BS(8) = BS(1) * BS(7)/BS(5)

```

EXPA 106
EXPA 107
EXPA 108
EXPA 109
EXPA 110
EXPA 111
EXPA 112
EXPA 113
EXPA 114
EXPA 115
EXPA 116
EXPA 117
EXPA 118
EXPA 119
EXPA 120
EXPA 121
EXPA 122
EXPA 123
EXPA 124
EXPA 125
EXPA 126
EXPA 127
EXPA 128
EXPA 129
EXPA 130
EXPA 131
EXPA 132
EXPA 133
EXPA 134
EXPA 135
EXPA 136
EXPA 137
EXPA 138
EXPA 139
EXPA 140

EXPAND

EXPAND

CP = (Z**6/(G-1.))=1.0) / (0.5*G*EM30)

RETURN

C

END

EXPA 141
EXPA 142
EXPA 143
EXPA 144
EXPA 145

EXPAND

CONE

```
001 CONE
002 CONE
003 CONE
004 CONE
005 CONE
006 CONE
007 CONE
008 CONE
009 CONE
010 CONE
011 CONE
012 CONE
013 CONE
014 CONE
015 CONE
016 CONE
017 CONE
018 CONE
019 CONE
020 CONE
021 CONE
022 CONE
023 CONE
024 CONE
025 CONE
026 CONE
027 CONE
028 CONE
029 CONE
030 CONE
031 CONE
032 CONE
033 CONE
034 CONE
035 CONE

SURROUTINE CONE (ANGLE,CP,ISDET)
C THIS IS AN IMPROVED APPROXIMATE CONE FLOW SOLUTION
C DEVELOPED BY D. N. SMYTH
C SECOND ORDER THEORY USED FOR SMALL VALUES OF SIMILARITY PARAMETER
C APPROXIMATE SOLUTION OF HAMMITT AND MURTHY FOR LARGE VALUES
C MIDRANGE GIVEN BY SUITABLE TRANSITION FUNCTION
C
COMMON /FSBS/FS,RS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
DIMENSION ANGLE(3),FS(A),BS(A)
DIMENSION TITLE(15)
REAL MACH
INTEGER CASE,PAGE,ERRR
DATA RC/,17453292E+1/
G = GAM
EM = FS(6)
ANGLE(2) = ARS(ANGLE(1))
DCR = ANGLE(2)*RC
C TEST FOR ZERO CONE ANGLE
IF (DCR .GT. 1.5-4) GO TO 7
C CONE ANGLE TOO SMALL. SET PROPERTIES AND RETURN.
CP = 0.0
DO 1 I = 1,6
1 RS(I) = FS(I)
6 ANGLE(3) = ARSIN(1.0/EM)/RC
RETURN
C
7 CONTINUE
SINDC = SIN(DCR)
FMSIN = EM*SINDC
EMSQ = EM**2
KSOL = 2
```

CONE

CONE

CONE 036
 CONE 037
 CONE 038
 CONE 039
 CONE 040
 CONE 041
 CONE 042
 CONE 043
 CONE 044
 CONE 045
 CONE 046
 CONE 047
 CONE 048
 CONE 049
 CONE 050
 CONE 051
 CONE 052
 CONE 053
 CONE 054
 CONE 055
 CONE 056
 CONE 057
 CONE 058
 CONE 059
 CONE 060
 CONE 061
 CONE 062
 CONE 063
 CONE 064
 CONE 065
 CONE 066
 CONE 067
 CONE 068
 CONE 069
 CONE 070

```

C
C
C   CALCULATE UPPER TRANSITION POINT, EMSINF,
      EMSINF = 1.40
      IF (FM.GE.10.0) GO TO A
      EMSINF = 0.325
      IF (FM.GT.1.5) EMSINF = 1.40 + 1.075*EXP((FM-1.5)*(-0.8))
      A IF (EMSIN.LT.EMSINF) GO TO 20
      KSOI = 1

C
C   CALCULATE CONE SURFACE PROPERTIES USING METHOD OF
C   HAMMITT AND MURTHY.
      9 SINQ = SINDC**2
      SIN2 = SIN(2.*DCR)
      H2 = (G+1.)*SINQ + 2./EMSQ
      H1 = 2. - (G+5.)*SINQ
      HX = H2+H1/SIN2**2
      IF (HX .GT. -1.0) GO TO 12

C
C   FLOW DETACHED. USE TAN-CONE EMPIRICAL
      CK = 2.*(G+1.)/(G+3.)*EMSIN
      DSR = (CK+EXP(-CK))/EM
      IF (DSR .GT. 1.0) DSR = 1.0
      DSR = ARSIN(DSR)
      KSOI = 1
      GO TO 11
12 CONTINUE
      IF (ABS(H1).LE.1.E-3) GO TO 10
      RAD = SQRT(1.0 + HX)
      DSR = DCR + SIN2*(1.0 - RAD)/H1
      GO TO 11
10 DSR = DCR + 0.5*H2/SIN2*(1.0 + 0.25*HX*(1.0 - 0.5*HAX))
11 EMNSSQ = EMSQ*SIN(DSR)**2
      HSP = (DSR - DCR)**2
      EMC = SQRT((FMSQ-EMNSSQ)*(1.0 + 2.0*HSP)/(1.0 + 0.5*(G - 1.0)*

```

CONE

CONE

```

1      (EMNSSQ*(1.0 + 2.0*MS2) - 2.0*MS2*EMSQ))
      FLIM = (G - 1.0)/(G + 1.0)
      PSP1 = 2.0*G/(G+1.0)*EMNSSQ + ELIM
      TITS = (PSP1 + ELIM)/(1.0 + ELIM*PSP1)/PSP1
      TCT1 = (1.0 + (G+1.0)*0.5*EMSQ)/(1.0 + (G-1.0)*0.5*EMC**2)
      PCPS = (TCT1*TITS)**(G/(G-1.0))
      PCP1 = PCPS*PSP1
      CP = 2.0*(PCP1 - 1.0)/(G*EMSQ)
      GO TO (40,31), KSOL

C
C
C      CALCULATE LOWER TRANSITION POINT, EMSINO,
20      EMSINO = 0.2
      IF (EM.GT.3.0) EMSINO = 0.3
      IF (FMSIN.GT.EMSINO) GO TO 30
      KSOL = 1

C
C      CALCULATE CONE SURFACE PROPERTIES USING 2ND-ORDER THEORY,
21      BETA = SQRT(EMSQ - 1.0)
      TANDC = TAN(DCR)
      TANSQ = TANDC**2
      BETA2 = 5.0*FMSQ - 1.0
      AM = 3.25*EMSQ + 0.5 + (G+1.0)*(EMSQ/BETA)**2
      PHI = 0.69314718 - ALOG(BETA) - ALOG(TANDC)
      CP = TANSQ*(2.*PHI-1.0)*TANSQ*(3.+(BETA*PHI)**2*BETA2*PHI + AM))
      IF (KSOL.EQ.2) GO TO 22
      DSR = ARSIN(1./EM)
      PCP1 = G*EMSQ*CP *0.5 + 1.0
      TCT1 = PCP1**((G+1.0)/G)
      EMC = SQRT(2./(G-1.0))*((1.0+(G-1.0)*0.5*EMSQ)/TCT1 - 1.0))
      GO TO 40

22      DCP2 = TANDC*(4.+(PHI - 1.0) + TANSQ*(4.+(PHI**2.+(PHI-1.0)*
      BETA**2 + BETA2*(4.+(PHI - 1.0) + 4.*AM)))/(EM*COS(DCR)**3)
      GO TO 32
C

```

CONE 071
CONE 072
CONE 073
CONE 074
CONE 075
CONE 076
CONE 077
CONE 078
CONE 079
CONE 080
CONE 081
CONE 082
CONE 083
CONE 084
CONE 085
CONE 086
CONE 087
CONE 088
CONE 089
CONE 090
CONE 091
CONE 092
CONE 093
CONE 094
CONE 095
CONE 096
CONE 097
CONE 098
CONE 099
CONE 100
CONE 101
CONE 102
CONE 103
CONE 104
CONE 105

CONE

COME

C CALCULATE CONE SURFACE PROPERTIES USING TRANSITION SOLUTION
C (THAT IS, A COMBINATION OF RUTH 2ND-ORDER AND HAMMITT=MURTHY).

```

30 DXF = EMSINF - EMSINO
   DX = EMSIN - EMSINO
   SINDC = FMSINF/EM
   DCP = ARSIN(SINDC)
   GO TO 9

31 CPHM = CP
   OCR = ARSIN(EMSINO/EM)
   GO TO 21

32 A = (CPHM - CP - DCP2*DXF)/DXF**2
   CP = CP + DX*(DCP2 + A*DX)
   PCPI = 0.5*G*EMSSQ*CP + 1.0
   FMNSF = EM*SIN(DSR)
   EMNS = (FMNSF - 1.)*(EMSTN/EMSINF)**2 + 1.0
   EMNSSQ = EMNS**2
   FMSSG = ((G+1.)*EM*EMNS)**2 + 4.*(EM*SSQ+1.)*(G*EMNSSQ+1.)
   DGR = ARSIN(FMNS/EM)
   FMSSQ = EMSSQ/((2.*G*EMNSSQ-(G-1.))*((G+1.)*FMNSSQ+2.))
   PSPC = (2.*G*FMNSSQ - (G-1.))/(G+1.)/PCPI
   EMC = SQRT((1.+0.5*(G-1.)*EMSSQ)*PSPC**((G-1.)/G) - 1.)*2/(G-1.)
   TCT1 = (1.+0.5*EMSSQ)/(1.+0.5*EMC**2)
40 CONTINUE

C CALCULATE FLOW ARRAY BS
  ANGLF(3) = DGR/RC
  BS(1) = FS(1)*PCPI/TCT1
  BS(2) = FS(2)*PCPI
  BS(3) = FS(3)*TCT1
  BS(4) = FS(4)*SQRT(TCT1)
  IF (BS(3).GE.AV1) BS(5) = 2.27E-2*BS(3)**1.5/(BS(3)+198.6)
  IF (BS(3).LT.AV1) BS(5) = AV2*BS(3)**AV3
  BS(6) = EMC
  BS(7) = BS(6)*BS(4)
  BS(8) = BS(1)*BS(7)/BS(5)

```

COME

CONE

CONE 141
CONE 142
CONE 143
CONE 144

C
C

RETURN
END

CONE

NEWTPM 001
 NEWTPM 002
 NEWTPM 003
 NEWTPM 004
 NEWTPM 005
 NEWTPM 006
 NEWTPM 007
 NEWTPM 008
 NEWTPM 009
 NEWTPM 010
 NEWTPM 011
 NEWTPM 012
 NEWTPM 013
 NEWTPM 014
 NEWTPM 015
 NEWTPM 016
 NEWTPM 017
 NEWTPM 018
 NEWTPM 019
 NEWTPM 020
 NEWTPM 021
 NEWTPM 022
 NEWTPM 023
 NEWTPM 024
 NEWTPM 025
 NEWTPM 026
 NEWTPM 027
 NEWTPM 028
 NEWTPM 029
 NEWTPM 030
 NEWTPM 031
 NEWTPM 032
 NEWTPM 033
 NEWTPM 034
 NEWTPM 035

```

SURROUTINE NEWTPM (ANGLE,EMN,CP,ETAC,IPRINT,HER,CPSTAG,
  1 ISF,IFIRST )
C THIS SUBROUTINE CALCULATES THE SURFACE CONDITIONS USING THE BLUNT
C BODY SHOCK-EXPANSION TECHNIQUE OF KAUFMAN, JOURNAL OF THE
C ASTRONAUTICAL SCIENCES, VOL X, NO.2 SUMMER 1963.
C
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
  1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /FSRS/FS,BS
DIMENSION FS(A),RS(8),ANGLE(3)
DIMENSION TITLE(15)
INTEGER CASE,PAGE,ERROR
INTEGER TAPEIN,TAPEOT,TAPFA,TAPEH,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
  1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C REAL MACH0, MACH, MSURG, MU, MACH90, M1, M2
C COMMON /GAS0/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
C
IF (CPSTAG .LE. 0.01) CPSTAG = 2.0
IF (ETAC .LE. 0.01) ETAC = 1.0
WER = 0
IF (IFIRST .EQ. 1) GO TO 60
IFIRST = 1
MACH0 = FS(6)
C THE FOLLOWING SECTION PERFORMS AN ITERATION TO FIND THE MATCHING
C POINT MACH NUMBER.
C
C SET GAMMA
G = GAM
EMLOW = 0.91 + 0.3125*G

```

NEWTPM

NEWT 036
 NEWT 037
 NEWT 038
 NEWT 039
 NEWT 040
 NEWT 041
 NEWT 042
 NEWT 043
 NEWT 044
 NEWT 045
 NEWT 046
 NEWT 047
 NEWT 048
 NEWT 049
 NEWT 050
 NEWT 051
 NEWT 052
 NEWT 053
 NEWT 054
 NEWT 055
 NEWT 056
 NEWT 057
 NEWT 058
 NEWT 059
 NEWT 060
 NEWT 061
 NEWT 062
 NEWT 063
 NEWT 064
 NEWT 065
 NEWT 066
 NEWT 067
 NEWT 068
 NEWT 069
 NEWT 070

```

EMUP = EMLOW + 0.40
JJ# 0
P2 = 0.0
M2 = 0.0
C SOLVE FOR PRESSURE RATIO (1.0/ER, 100 OF TR 1135)
PCAP# (2.0/((G+1.0)*MACHO*MACHO))*G/(G+1.0) *
1 ((2.0*G*MACHO*MACHO/(G+1.0))/(G+1.0))*G/(G+1.0)
C ASSUME MSUBQ = EMLOW
MSUBQ = EMLOW
C CALCULATE Q (SEE KAUFMAN)
10 Q = (2.0/(2.0+(G+1.0)*MSUBQ*MSUBQ))*G/(G+1.0)
C CALCULATE P SUB C (EQ 9 OF KAUFMAN)
PC = Q * (1.0 + (G*MSURQ**4)/ (4.0+(MSUBQ*MSUBQ+1.0)*(1.00)))
CPO = (2.0 / (G*MACHO*MACHO)) * (Q/PC + 1.0)
IF (IPRINT, NE, 1) GO TO 30
WRITE (TAPEUT, 20) MSUBQ, PC, CPO
20 FORMAT (1H, 17X7HMSUBQ =F9.6,6H PC =1PE11.4,7H CPO =E11.4 )
C CHECK ITERATION ACCURACY
30 IF (ABS(MSUBQ-M2) ,LT, 0.0001) GO TO 50
C SET UP ITERATION TERMS
P1 = P2
P2 = PC
M1 = M2
M2 = MSUBQ
C STEP ITERATION COUNTER AND CHECK CYCLE
JJ# JJ# + 1
IF (JJ.GT. 1) GO TO 40
MSUBQ = EMUP
GO TO 10
  
```

NEWTPM

```

C 40 IF (JJ.GT.20) GO TO 50
C ESTIMATE NEW M
  IF (ABS(P2-P1) .LT. 0.000001) GO TO 50
  MSUBQ = M1 + (PCAP-P1)*(M2-M1)/(P2-P1)
C CHECK NEW ESTIMATE FOR M
  IF (MSUBQ .GT. EMUP) MSUBQ = EMUP
  IF (MSUBQ .LT. EMLDW) MSUBQ = EMLDW
  GO TO 10
C
C CALCULATE MATCHING POINT IMPACT ANGLE
50 IF ((G-P) .LE. 0.0) GO TO 100
  SDELTA = SQRT((O-PCAP)/(1.0-PCAP))
  DELTA = ARSIN(SDELTA) * 0.5729578E02
C
C CALCULATE EXPANSION ANGLE FROM MATCHING POINT
60 DLTMU = DELTA = ANGLE(2)
C
C CHECK IF FLOW WILL EXPAND AT LEAST TO MATCHING POINT
  IF (DLTMU .LT. 0.0) GO TO 100
C
C DETERMINE MACH NUMBER ON SURFACE
  FS(6) = MSUBQ
  ANGLF(2) = DLTMU
C
  ISDET = 0
  CALL EXPAND (ANGLE,MFR,IPRINT,ISDET,CP)
  FS(6) = MACHN
C SET UP SURFACE MACH NUMBER
  MACH = BS(6)
C
C CALCULATE SURFACE PRESSURE RATIO (EQ. 4# OF TR 1135)
  PPO = ETAC * (1.0 + (G-1.0)*MACH*MACH/2.0)**(-G/(G-1.0))
C
C CALCULATE P / P FREE STREAM
  PPRS = (1.0/PCAP)*PPO

```

```

C CALCULATE PRESSURE COEFFICIENT ON SURFACE
CP = (2.0/(G*MACHO*MACHO))* (PPFS - 1.0)
IF (IPRINT, NE, 1) GO TO 80
WRITE (TAPEOT, 70) MSUBO, PCAP, Q, PPO, MACH, DELTQ, PC, DLTMU, PPFS, CP
70 FORMAT (1H, 17X23HSHOCK-EXPANSION M G =F7,5,8H P CAP=IPE11.4,
1 RM Q =E11.4,8H P/PO =E11.4,7H MACH=OPF7.3,1H, 17X
2 23HCALCULATIONS DELTQ=F7,3,8H P C =IPE11.4, 8H DLTMU=
3 F11.4,8H P/PPFS=E11.4,5H CP=OPF9.5 )
C CHECK IF FLOW CONDITIONS ARE NEEDED
80 IF (ISE .GT. 0) GO TO 110
C
C CALCULATE FREE STREAM TOTAL TEMPERATURE (EQ. 43 OF TR 1135)
TSUBT = FS(3)*(1.0 + (G-1.0)*MACHO*MACHO/2.0)
C
C CALCULATE TEMPERATURE AFTER EXPANSION (IN RANKINE)
Y = TSUBT / (1.0 + (G-1.0)*MACH*MACH / 2.0)
C
C CALCULATE SURFACE PRESSURE
P = PPFS * FS(2)
C
C CALCULATE DENSITY (EQ. 26 OF TR 1135)
RHO = GAP/((G-1.)*GASCPT)
C
C CALCULATE LOCAL SPEED OF SOUND
A = SQRT(G*P/RHO)
C
C CALCULATE LOCAL VELOCITY
V = MACH * A
C
C CALCULATE VISCOSITY MU = 2.27E-8*Y**1.5/(T+198.6)
IF(T,GE,AV1)
IF(T,LT,AV1) MU = AV2*T**AV3
C
C CALCULATE REYNOLDS NUMBER PER FOOT (EQ) BY TR1135)
RE = RHO * V / MU

```

NEWT 106
 NEWT 107
 NEWT 108
 NEWT 109
 NEWT 110
 NEWT 111
 NEWT 112
 NEWT 113
 NEWT 114
 NEWT 115
 NEWT 116
 NEWT 117
 NEWT 118
 NEWT 119
 NEWT 120
 NEWT 121
 NEWT 122
 NEWT 123
 NEWT 124
 NEWT 125
 NEWT 126
 NEWT 127
 NEWT 128
 NEWT 129
 NEWT 130
 NEWT 131
 NEWT 132
 NEWT 133
 NEWT 134
 NEWT 135
 NEWT 136
 NEWT 137
 NEWT 138
 NEWT 139
 NEWT 140

NEWTPM

```

C SET UP DATA FOR USE BACK IN OTHER SUBROUTINES
  RS(1) = RHO
  BS(2) = P
  BS(3) = T
  BS(4) = A
  RS(5) = MU
  RS(6) = MACH
  RS(7) = V
  RS(8) = RE
  IF (IPRINT .NE. 1) GO TO 110
  WRITE (TAPEOT,90) MACH,A,RF,P,TSUBT,V,T,RHO,MU
  90 FORMAT (1H,34X6MACH #F7.5,8H A =1PE11.4,8H RE #E11.4
    1 8H P #E11.4,7H TTOT30PF7.1,1H,34X6MV #E11.4
    2 1PE11.4,8H RHO #E11.4,8H MU #E11.4 )
C
C GO TO 110
C
C FLOW WAS NOT REACHED THE MATCHING POINT, USE NEWTONIAN CALCULATIONS
C AND SHOCK DETACHED METHOD SUGGESTED BY SMYTH.
  100 CP = CPSTAG * (SIN(ANGLE(2))/572957795E+02)**2
C
C CHECK IF FLOW CONDITIONS ARE NEEDED
  IF (ISE .GT. 0) GO TO 110
  MFR = 1
C
C CALCULATE FLOW DATA FOR DETACHED CONDITIONS
C CALCULATE SQUARE OF MACH NUMBER NORMAL TO EFFECTIVE SHOCK
  MACHSQ=(MACH0+MACH0*CP*(4./(G+1.)))/(2.*(1.-(G+1.)))/(G+1.))
C CALCULATE EFFECTIVE DENSITY RATIO
  EPSI = ((G+1.0)/(G+1.0)) * (1.0 + 2.0/(G+1.0))*MACHSQ)
C CALCULATE THE EFFECTIVE SHOCK ANGLE SQUARED
  ANGLE(3) = CP / (2.0*(1.0-EPSI))

```

NEWT 141
 NEWT 142
 NEWT 143
 NEWT 144
 NEWT 145
 NEWT 146
 NEWT 147
 NEWT 148
 NEWT 149
 NEWT 150
 NEWT 151
 NEWT 152
 NEWT 153
 NEWT 154
 NEWT 155
 NEWT 156
 NEWT 157
 NEWT 158
 NEWT 159
 NEWT 160
 NEWT 161
 NEWT 162
 NEWT 163
 NEWT 164
 NEWT 165
 NEWT 166
 NEWT 167
 NEWT 168
 NEWT 169
 NEWT 170
 NEWT 171
 NEWT 172
 NEWT 173
 NEWT 174
 NEWT 175

NEWTPM

NEWTPM

NEWT	176
NEWT	177
NEWT	178
NEWT	179
NEWT	180
NEWT	181

C CALCULATE NORMAL MACH SQUARED TIMES SQUARE OF SHOCK ANGLE
PMN = MACHSQ * ANGLE(3)

C 110 RETURN

C END

NEWTPM

ATMOS

001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035

SUBROUTINE ATMOS (A3,A8,A4,A1,A6)
 THIS ROUTINE CALCULATES ATMOSPHERIC PROPERTIES OF THE
 US STANDARD ATMOSPHERE, 1962, ASSUMING AN INVERSE SQUARE
 GRAVITATIONAL FIELD. THIS ASSUMPTION YIELDS DATA THAT
 AGREES WITH THE COESA DOCUMENT WITHIN 1 PER CENT AT
 ALL ALTITUDES UP TO 700 KILOMETERS (226558 FEET). THE
 DATA IS ARRANGED IN THE ATMOSPHERE ARRAY, A, AS
 FOLLOWS
 A(1) = CS, SPEED OF SOUND, F./SEC
 A(2) = (1/CS)(DCS/DZ), SOUND DERIVATIVE, 1/FT
 A(3) = Z, GEOMETRIC ALTITUDE, FT (GIVFN)
 A(4) = P, PRESSURE, LB/FT2
 A(5) = DP/DZ, PRESSURE DERIVATIVE, LB/FT3
 A(6) = RHO, DENSITY, SLUGS/FT3
 A(7) = (1/RHO)(DRHO/DZ), DENSITY DERIVATIVE, 1/FT
 A(8) = T, TEMPERATURE, DEG RANKINE
 A(9) = DT/DZ, TEMPERATURE DERIVATIVE, DEG RANKINE/FT
 VARIOUS CONSTANTS USED
 EARTH RADIUS = 20890855 FT
 SPECIFIC HEAT RATIO FOR AIR = 1.4
 SEA LEVEL VALUES
 GRAVITATIONAL ACCELERATION = 32.1740484 FT/SEC2
 MOLECULAR WEIGHT = 28.9644
 GM*MO/R* = 0.016743418 DEG RANK/FT
 DIMENSION A(9),HG(10),ZM(16),MM(14),TM(23),PM(22)
 SET ARRAYS AND CONSTANT VALUES
 DATA GO,MMO,RO,GHRS/32,1740484,28.9644,20890855.0,
 1 0.016743418,16404.00
 2 ,36089.,65617.,104987.,154199.,170604.,200131.,
 3 259186.,291160.,7M/295276.,328084.,
 4 360892.,393701.,492126.,524934.,557743.,623360.,

ATMOS

ATM08

```

5 754593.,984252.,1312336.,1640420.,1968504.,
6 2296588./,MM/28,9644,28.88,29.56,
7 28.07,26.92,26.66,26.4,25.85,24.7,22.66,19.94,
8 17.94,16.64,16.17/

DATA TM/577,17,518,67,389,97,389,97,411,57
1 ,487,17,487,17,454,77,325,17,325,17,379,17,469,17
2 ,649,17,1729,17,1999,17,2179,17,2431,17,2791,17
3 ,3295,17,3889,17,4357,17,4861,17,5461,17/,PM/
4 3711,0839,2116,2165,472,67563,114,34314,
5 18,128355,2,3162178,1,2321972,3,8030279E-01,
6 2,1671352F-02,3,4313478E+03,6,2773411F-04,1,53490
7 91E-04,5,2624212E-05,1,0561806E-05,7,7083076E-06,
8 5,8267151F-06,3,3159854E-06,1,4520255E-06,3,92905
9 63E-07,8,4030242E-06,2,2835256E-08,7,1875452E-09/

```

A(3) = A3

C CALCULATE G, Z, AND CHECK

10 Z = A(3)

G = G0*(R0/(R0+Z))**2

IF (Z .GT. 295276.0) GO TO 50

C TMS LINEAR WITH GEOPOTENTIAL. CALCULATE H AND SEARCH

H = R0*Z/(R0+Z)

DO 20 I = 2,10

J = I - 1

IF (MG(I) .GE. H) GO TO 30

20 CONTINUE

C CALCULATE TMS SLOPE, TMS, AND SET MOL WT STUFF

30 ELH = (TM(J+1) - TM(J))/(MG(J+1) - MG(J))

TMS = TM(J) + FLH*(H - MG(J))

FLZ = ELH*G/G0

DMNZ = 0.0

ATMO 036
 ATMO 037
 ATMO 038
 ATMO 039
 ATMO 040
 ATMO 041
 ATMO 042
 ATMO 043
 ATMO 044
 ATMO 045
 ATMO 046
 ATMO 047
 ATMO 048
 ATMO 049
 ATMO 050
 ATMO 051
 ATMO 052
 ATMO 053
 ATMO 054
 ATMO 055
 ATMO 056
 ATMO 057
 ATMO 058
 ATMO 059
 ATMO 060
 ATMO 061
 ATMO 062
 ATMO 063
 ATMO 064
 ATMO 065
 ATMO 066
 ATMO 067
 ATMO 068
 ATMO 069
 ATMO 070

ATM08

```

EM      = WMO
C CHECK TMS SLOPE AND CALCULATE PRESSURE
  IF (ELH .EQ. 0.0) GO TO 40
C NON = ZERO SLOPE PRESSURE EQUATION
  A(4) = PM(J)*(TM(J)/TMS)**(GMRS/ELH)
  GO TO 80
C ZERO SLOPE PRESSURE EQUATION
  40 A(4) = PM(J)*EXP(GMRS*(HG(J)-H)/TMS)
  GO TO 80
C TMS LINEAR WITH Z, SEARCH MATRIX
  50 DO 60 I = 2,14
    J = I + 8
    K = I - 1
    IF (ZM(I) .GE. Z) GO TO 70
    60 CONTINUE
C CALCULATE TMS, SLOPE, AND STUFF
  70 ELZ = (TM(J+1) - TM(J))/(ZM(K+1) - ZM(K))
    TMS = TM(J) + ELZ*(Z - ZM(K))
    DMDZ = (WM(K+1) - WM(K))/(ZM(K+1) - ZM(K))
    EM = WM(K) + DMDZ*(Z - ZM(K))
    ZLZ = Z - TMS/ELZ
C PRESSURE EQUATION FOR TMS LINEAR WITH Z
  A(4) = PM(J)*EXP(GMRS/ELZ*(R0/(R0+ZLZ))**2*((Z-ZM(K)))*
1 (R0+ZLZ)/(R0+Z)/(R0+ZM(K)) - ALNG(TMS*(R0+ZM(K))
2 )/TM(J)/(R0+Z)))
C CALCULATE SOUND SPEED AND DERIVATIVE
  80 A(1) = 49.022164*SQRT(TMS)
    A(2) = 0.5*ELZ/TMS

```

ATMOS

ATMO 106
ATMO 107
ATMO 108
ATMO 109
ATMO 110
ATMO 111
ATMO 112
ATMO 113
ATMO 114
ATMO 115
ATMO 116
ATMO 117
ATMO 118
ATMO 119
ATMO 120

C CALCULATE DENSITY, DERIVATIVE, AND PRESSURE DERIVATIVE
A(6) = GMRS*A(4)/G0/TMS
A(7) = (A(6)*G/A(4) + ELZ/TMS)
A(5) = A(6)*G

C CALCULATE TEMPERATURE, DERIVATIVE, AND LEAVE
A(8) = EM*TMS/WMO
A(9) = (EM*ELZ + TMS*DMDZ)/WMO
90 A8 = A(8)
A4 = A(4)
A1 = A(1)
A6 = A(6)
RETURN
END

ATMOS

```

C C SURROUTINE SOLVIT(A,ND,WD,KD,NI,MM,ND,NM,8B,NMB,NMB,NERR)
C C
C C      *** / ***** ***/ * ***/
C C      * * / * * * * * * / * * ***/
C C      **** * / * * * * * * / * * ****
C C      * * / * * * * * * / * *
C C      * / *** * * * * * / ***
C C
C C      D I R E C T M A T R I X S O L U T I O N
C C
C C      WRITTEN BY J. L. HESS * PROGRAMMED BY " . M. RIDDELL
C C
C C      THIS VERSION MODIFIED TO DO PIVOTAL SEARCH BY ROWS.
C C      REFER TO PAGE 35 OF DAC-7000 FOR DESCRIPTION OF METHOD.
C C      CODED BY D. N. SMYTH
C C
C C      DIMENSION A ( KD ), IRC(500), ISC(500)
C C      AA IS USED WHEN SOLUTIONS ARE WRITTEN ON TAPE NM.
C C      DIMENSION AA(500)
C C      BB IS USED WHEN SOLUTIONS ARE RETURNED THROUGH CALL STATEMENT.
C C      DIMENSION BB(NMB,NMB)
C C
C C      LOGICAL LAST
C C
C C      NERR = 0
C C      N = ND
C C      M = WD
C C      KORE = KD
C C      NPM = N + M
C C      IF (MAX0(3 * YPM, M * N) .GT. KORE) STPP 1
C C      MT = MM
C C      PEWIND MT
C C      NIN = NI

```

SOLV 001
SOLV 002
SOLV 003
SOLV 004
SOLV 005
SOLV 006
SOLV 007
SOLV 008
SOLV 009
SOLV 010
SOLV 011
SOLV 012
SOLV 013
SOLV 014
SOLV 015
SOLV 016
SOLV 017
SOLV 018
SOLV 019
SOLV 020
SOLV 021
SOLV 022
SOLV 023
SOLV 024
SOLV 025
SOLV 026
SOLV 027
SOLV 028
SOLV 029
SOLV 030
SOLV 031
SOLV 032
SOLV 033
SOLV 034
SOLV 035

SOLVIT

```
REWIND NIN
NOUT = NO
REWIND NOUT
MPI = M + 1
NN = N
NEL = NPM

C INITIALIZE INTERCHANGE COUNTERS
DO 5 I = 1, N
  IRC(I) = 0
  JSC(I) = I
5 CONTINUE
IR = 0
INT = 0
NM1 = N - 1

C - - CALCULATE THE MAXIMUM NO. OF ROWS, #K#
C
C 10 K = (KORE - NEL) / NEL
C - - TEST TO SEE IF THE REST OF THE MATRIX WILL FIT IN CCME
C
  LAST = K .GE. NM
  IF (LAST) K = NM
  IRN = IR

C - - READ #K# ROWS OF THE AUGMENTED #A# MATRIX
C
  20 NT = 0
  DO 30 IB = 1, K
    NS = NT + 1
    NT = NT + NEL
  30 CALL GETT(NIN, 1, NEL, A(NS, 1, AA2))

C - - CHECK TO SEE IF WE WERE UNLUCKY ENOUGH TO END UP WITH ONLY ONE ROW
C
```

SOLV 036
SOLV 037
SOLV 038
SOLV 039
SOLV 040
SOLV 041
SOLV 042
SOLV 043
SOLV 044
SOLV 045
SOLV 046
SOLV 047
SOLV 048
SOLV 049
SOLV 050
SOLV 051
SOLV 052
SOLV 053
SOLV 054
SOLV 055
SOLV 056
SOLV 057
SOLV 058
SOLV 059
SOLV 060
SOLV 061
SOLV 062
SOLV 063
SOLV 064
SOLV 065
SOLV 066
SOLV 067
SOLV 068
SOLV 069
SOLV 070

SOLVIT

```

C      IF (K .EQ. 1) GO TO 90
C
C      - - - - - IS GREATER THAN #1# SO WE CAN START THE TRIANGULARIZATION
C
      NELP1 = NEL + 1
      NS = NEL
      NELP2 = NELP1 + 1
C
C      - - - - - FORM THE #TRAPPEZOIDAL# ARRAY (8)
C
      K1 = K
      IF (K .LT. NV) K1 = K1 + 1
      DO 40 JB = 2, K1
      NP = NELP2 - JB
      NS = NS + NELP1
      NT = NS
C
C      SEARCH FOR PIVOTAL ELEMENT, AMAX
      IMP = 0
      AMAX = 0.0
      I = IR
      IR = IR + 1
      NN = NS + N - IP
      DO 35 I2 = NS, NN
      I = I + 1
      IF (ABS(A(I2)) .LT. ABS(AMAX)) GO TO 35
      AMAX = A(I2)
      IMA = I2
      IMR = I
C
C      35 CONTINUE
C      SFT INTERCHANGE COUNTERS AND INTERCHANGE ELEMENTS IN BASE ROW
      IF (IMP .EQ. 0) GO TO 36
      IRC (IP) = IMR
      A(IMA) = A(NS)

```

```

SOLV 071
SOLV 072
SOLV 073
SOLV 074
SOLV 075
SOLV 076
SOLV 077
SOLV 078
SOLV 079
SOLV 080
SOLV 081
SOLV 082
SOLV 083
SOLV 084
SOLV 085
SOLV 086
SOLV 087
SOLV 088
SOLV 089
SOLV 090
SOLV 091
SOLV 092
SOLV 093
SOLV 094
SOLV 095
SOLV 096
SOLV 097
SOLV 098
SOLV 099
SOLV 100
SOLV 101
SOLV 102
SOLV 103
SOLV 104
SOLV 105

```

SOLVIT

SOLV 106
 SOLV 107
 SOLV 108
 SOLV 109
 SOLV 110
 SOLV 111
 SOLV 112
 SOLV 113
 SOLV 114
 SOLV 115
 SOLV 116
 SOLV 117
 SOLV 118
 SOLV 119
 SOLV 120
 SOLV 121
 SOLV 122
 SOLV 123
 SOLV 124
 SOLV 125
 SOLV 126
 SOLV 127
 SOLV 128
 SOLV 129
 SOLV 130
 SOLV 131
 SOLV 132
 SOLV 133
 SOLV 134
 SOLV 135
 SOLV 136
 SOLV 137
 SOLV 138
 SOLV 139
 SOLV 140

```

C
A(NS) = AMAX
IS = ISC(IR)
ISC(IR) = ISC(IMR)
ISC(IMR) = IS

C
36 CONTINUE
IF (IB .GT. K) GO TO 45
DO 40 IO = 1, K
  NT = NT + NEL
  MN = NT
  NB = NS

C
C INTERCHANGE ELEMENTS IN OBJECT ROW
IMA = IMA + NEL
AMAX = A(IMA)
A(IMA) = A(NT)
A(NT) = -AMAX/A(NS)
DO 40 NF = 2, NP
  MN = MN + 1
  NB = NB + 1
40 A(MN) = A(MN) + A(NT) * A(NB)
45 CONTINUE
IF (LAST) GO TO 90

C
C * = WRITE THE TRAPEZOIDAL MATRIX ON TAPE
C
NT = 0
NP = NEL
NS = -NFL
DO 50 IO = 1, K
  NS = NS + NELPI
  NT = NT + NEL
  NN = NS + NP + 1
CALL SAVE(MT, 2, NP, NP, A(NS), 1, AAP)

```

SOLVIT

SOLVIT

```
50 NP = NF + 1
   NP = NP + M
   NS = KORE + NEL + 1
C - - READ ANOTHER ROW
C
   DO 60 IC = 1, NP
   CALL GETT(NIN, I, NEL, A(NS), I, AA2)
C - - MODIFY THIS ROW BY THE TRAPEZOIDAL ARRAY
C
   IR = IRO
   NT = 1
   MN = NS
   DO 70 IB = 1, K
   NB = NT
   NF = MN + 1
   IR = IR + 1
   I2 = NS + IRC(IR) + 1 - IRO
   A(I2) = A(I2)
   A(MN) = A(MN)
   A(MN) = -AMAX/A(NT)
   DO 60 NN = NF, KORE
   NB = NB + 1
   60 A(NN) = A(NN) + A(MN) * A(NB)
   MN = NF
   70 NT = NT + NELP1
C - - WRITE THE MODIFIED ROW ON TAPE
C
   NNI = KORE + MN + 1
   80 CALL SAVE(NOUT, I, NNI, NN1, A(MN), I, AA2)
   REWIND NOUT
   REWIND NIN
C
```

```
SOLV 141
SOLV 142
SOLV 143
SOLV 144
SOLV 145
SOLV 146
SOLV 147
SOLV 148
SOLV 149
SOLV 150
SOLV 151
SOLV 152
SOLV 153
SOLV 154
SOLV 155
SOLV 156
SOLV 157
SOLV 158
SOLV 159
SOLV 160
SOLV 161
SOLV 162
SOLV 163
SOLV 164
SOLV 165
SOLV 166
SOLV 167
SOLV 168
SOLV 169
SOLV 170
SOLV 171
SOLV 172
SOLV 173
SOLV 174
SOLV 175
```

SOLVIT

SOLVIT

SOLV 176
SOLV 177
SOLV 178
SOLV 179
SOLV 180
SOLV 181
SOLV 182
SOLV 183
SOLV 184
SOLV 185
SOLV 186
SOLV 187
SOLV 188
SOLV 189
SOLV 190
SOLV 191
SOLV 192
SOLV 193
SOLV 194
SOLV 195
SOLV 196
SOLV 197
SOLV 198
SOLV 199
SOLV 200
SOLV 201
SOLV 202
SOLV 203
SOLV 204
SOLV 205
SOLV 206
SOLV 207
SOLV 208
SOLV 209
SOLV 210

SOLVIT

```
C - - SWITCH THE TAPES
C
  NT = NIN
  NIN = NOUT
  NOUT = NT
C - - RE-CALCULATE ROW LENGTH AND LOOP BACK
C
  NEL = NEL * K
  NN = NEL * M
  GO TO 10
C - - REWIND ALL TAPFS
C
  90 REWIND MT
  REWIND NIN
  REWIND NOUT
C
  C HAVE K ROWS OF TRAPEZOIDAL ARRAY IN CORE,
  C PERFORM COLUMN INTERCHANGES UN PRECEDING ROWS
  C DUE TO FOLLOWING ROWS
  C
    IF (K .LT. 3) GO TO 96
    NN = K * 2
    NS = -NEL
    DO 95 IB = 1, NN
    NS = NS + NELP1
    I1 = IB + IRN + 1
    DO 94 IR = I1, NM1
    IF (IRC(IR) .EQ. IR) GO TO 94
    I2 = IRC(IR) * IRO
    IF (I2 .LE. 0) GO TO 94
    I2 = I2 + NS - IB
    I3 = IR + IRO + NS - IB
    AMIX = A(I2)
```

SOLVIT

SOLV 211
SOLV 212
SOLV 213
SOLV 214
SOLV 215
SOLV 216
SOLV 217
SOLV 218
SOLV 219
SOLV 220
SOLV 221
SOLV 222
SOLV 223
SOLV 224
SOLV 225
SOLV 226
SOLV 227
SOLV 228
SOLV 229
SOLV 230
SOLV 231
SOLV 232
SOLV 233
SOLV 234
SOLV 235
SOLV 236
SOLV 237
SOLV 238
SOLV 239
SOLV 240
SOLV 241
SOLV 242
SOLV 243
SOLV 244
SOLV 245

SOLVIT

```
A(I2) = A(I3)
A(I3) = AMAX
94 CONTINUE
95 CONTINUE
C
C - - CONDENSE THE MATRIX
C
96 MN = NEL
NL = NEL + 1
IF (K.EQ.1) GO TO 110
NS = 1
NT = NEL
DO 100 IR = 2, K
NS = NS + MELP1
NT = NT + NEL
DO 100 ID = NS, NT
A(NL) = A(ID)
100 NL = NL + 1
110 N1 = KORE = K * M + 1
C
C - - THERE, NOW WE CAN START THE BACK-SOLUTION
C * * NOTE..THE FIRST AVAILABLE LOCATION FOR THE SOLUTIONS IS A(N1)
C
NREM = N
NEL = NPM
LAST = K.EQ. N
NPASS = 0
C
C - - SOLVE FOR THE ANSWERS CORRESPONDING TO *K* ROWS
C
120 KM1 = K - 1
KPI = K + 1
NS = NL - MPI
NPASS = NPASS + 1
DO 150 MN = 1, M
```

SOLVIT

SOLV 246
SOLV 247
SOLV 248
SOLV 249
SOLV 250
SOLV 251
SOLV 252
SOLV 253
SOLV 254
SOLV 255
SOLV 256
SOLV 257
SOLV 258
SOLV 259
SOLV 260
SOLV 261
SOLV 262
SOLV 263
SOLV 264
SOLV 265
SOLV 266
SOLV 267
SOLV 268
SOLV 269
SOLV 270
SOLV 271
SOLV 272
SOLV 273
SOLV 274
SOLV 275
SOLV 276
SOLV 277
SOLV 278
SOLV 279
SOLV 280

```
NF = NS + MN
A(NF) = A(NF) / A(NS)
NT = NS
IF (KMI .EQ. 0) GO TO 150
DO 140 IB = 1, KMI
NF = NF + IB * M
NT = NT + MP1 * IB
SUM = 0.0
NP = NF
N2 = MP1 + IR
DO 130 IO = 1, IB
NN = NT + IO
NP = NP + N2 * IO
130 SUM = SUM + A(NN) * A(NP)
140 A(NF) = (A(NF) + SUM) / A(NT)
150 CONTINUE
C -- MOVE THE SOLUTIONS TO CONTIGUOUS LOCATIONS STARTING AT A(N1)
C
N1 = KORE + 1
DO 170 NN = 1, K
DO 160 MN = 1, M
NL = NL + 1
N1 = N1 + 1
160 A(N1) = A(NL)
170 NL = NL + MN
C -- WRITE THE SOLUTIONS ON TAPE
C
WRITE (NIN) K
NS = N1 + 1
DO 180 MN = 1, M
NT = NS + MN
180 WRITE (NIN) (A(IO), IO = NT, KORE, M)
C
```

SOLVIT

```

C - - TEST IF THIS IS THE LAST PASS
C
C   IF (LAST) GO TO 260
C - - WE MUST NOW MODIFY THE TRIANGULAR MATRIX TO REFLECT THE EFFECT OF
C   THE SOLUTIONS OBTAINED SO FAR (EQ 21)
C * * NOTE. LOCATIONS A(I) TO A(NI-1) ARE NOW FREE TO USE
C
C - - CALCULATE THE NEXT VALUES OF #NEL# AND #NREM#
C
      NELD0 = NEL
      KOLD = K
      NEL = NEL + K
      NREM = NREM + K
C
C ***** CALCULATE NEW K, B AND C (RFAL) WILL ALWAYS BE INTEGERS.
C   K WILL BE CALCULATED REAL AND TRUNCATED - - GOOD.
C
      R = 1 + 2*M
      C = 2*(KOLD*(M+1) + KORE)
      K = (-R + SQRT(B**2 + 4*C))/2.0
      NROW = NREM + K + 1
      IF (K .LT. NREM) GO TO 190
      LAST = .TRUE.
      NROW = 1
      K = NREM
      190 NS = 1
          NT = NELD0 + 1
          NP2 = N - 1
C - - READ IN THE ROWS TO BE MODIFIED
C
      NN = NPM
      DO 250 IB = 1, NREM
          NT = NT + 1

```

```

SOLV 261
SOLV 262
SOLV 283
SOLV 284
SOLV 285
SOLV 286
SOLV 287
SOLV 288
SOLV 289
SOLV 290
SOLV 291
SOLV 292
SOLV 293
SOLV 294
SOLV 295
SOLV 296
SOLV 297
SOLV 298
SOLV 299
SOLV 300
SOLV 301
SOLV 302
SOLV 303
SOLV 304
SOLV 305
SOLV 306
SOLV 307
SOLV 308
SOLV 309
SOLV 310
SOLV 311
SOLV 312
SOLV 313
SOLV 314
SOLV 315

```

SOLVIT

SOLV 316
 SOLV 317
 SOLV 318
 SOLV 319
 SOLV 320
 SOLV 321
 SOLV 322
 SOLV 323
 SOLV 324
 SOLV 325
 SOLV 326
 SOLV 327
 SOLV 328
 SOLV 329
 SOLV 330
 SOLV 331
 SOLV 332
 SOLV 333
 SOLV 334
 SOLV 335
 SOLV 336
 SOLV 337
 SOLV 338
 SOLV 339
 SOLV 340
 SOLV 341
 SOLV 342
 SOLV 343
 SOLV 344
 SOLV 345
 SOLV 346
 SOLV 347
 SOLV 348
 SOLV 349
 SOLV 350

SOLVIT

```

IF (IB .LE. NR0W) GO TO 200
NS = NS + NN
NT = NT + NN
200 CALL GET7(MT, 2, NN, A(NS), 1, AA2)
C
C THE FIRST TIME THE TRAPEZOIDAL ARRAY IS READ BACK,
C THE COLUMN ORDER IS MADE CONSISTENT.
C THAT IS, THE COLUMN INTER CHANGES OF FOLLOWING
C ROWS ARE MADE
C
IF (INT .NE. 0) GO TO 205
NR1 = IB + 1
DO 204 IR = NR1, NR2
IF (IRC(IR) .EQ. IR) GO TO 204
I2 = IRC(IR) + NS + IB
I3 = IR + IB + NS
AMAX = A(I2)
A(I2) = A(I3)
A(I3) = AMAX
204 CONTINUE
205 CONTINUE
NP = NI + 1
NF = NT + M - KM1
NN = NN - KOLD
DO 220 MN = 1, M
N2 = NF
NA = NP + MN
NB = NA
SUM = 0.0
DO 210 IO = 1, KOLD
SUM = SUM + A(N2) + A(NA)
N2 = N2 + 1
210 NA = NA + M
N2 = N2 + "N + 1
220 A(N2) = A(N2) + SUM
  
```

SOLVIT

```
C - - WRITE THE MODIFIED ROW ON TAPE OR CONDENSE THE ROW
C
C
NL = NT = M + 1
IF (IB .GE. NROW) GO TO 230
NF = NL = KP1
NN1 = NF = NS + 1
NN2 = NT = NI + 1
CALL SAVE(NOUT, 4, NN, NN1, A(NS), NN2, A(NL))
GO TO 250
230 NF = NL = KOLD
DO 240 MN = NL, NT
A(MN) = A(MN)
240 NF = NF + 1
250 CONTINUE
INT = 1
REWIND MT
REWIND NOUT
C - - SWITCH THE TAPE
C
C
NT = MT
MT = NOUT
NOUT = NT
C - - LOOP BACK THRU THE SOLUTION
C
C
NL = NF
GO TO 120
C - - START TO WRAP IT UP
C
C
260 REWIND NIN
N2 = N
C
```

SOLV 351
SOLV 352
SOLV 353
SOLV 354
SOLV 355
SOLV 356
SOLV 357
SOLV 358
SOLV 359
SOLV 360
SOLV 361
SOLV 362
SOLV 363
SOLV 364
SOLV 365
SOLV 366
SOLV 367
SOLV 368
SOLV 369
SOLV 370
SOLV 371
SOLV 372
SOLV 373
SOLV 374
SOLV 375
SOLV 376
SOLV 377
SOLV 378
SOLV 379
SOLV 380
SOLV 381
SOLV 382
SOLV 383
SOLV 384
SOLV 385

SOLVIT

SOLVIT

```

C * * NOTE.: AT THIS POINT ALL LOCATIONS A(I) THRU A(KORE) ARE FREE.
C
      DO 260 I6 = 1, NPASS
      READ (NIN) K
      N1 = N2 * K + 1
      NS = N1
      NT = N2
C * * READ IN THE SOLUTIONS
C
      DO 270 IO = 1, M
      NM = NT * NS + 1
      CALL GETT(NIN, 1, NM, A(NS), 1, AA2)
      NT = NT + N
      270 NS = NS + N
      280 N2 = N1 + 1
C * * WRITE THE SOLUTIONS ON TAPE
C
      NT = 0
      DO 290 IO = 1, M
      NS = NT + 1
      NT = NT + N
C ALL N+M SOLUTIONS ARE IN CORE. CORRECT ORDER
C DUE TO INTERCHANGES MADE DURING SOLUTION.
C
      DO 289 I = 1, N
      I1 = ISC(I)
      I2 = NS + I + 1
      AA(I1) = A(I2)
      BB(I1, IO) = A(I2)
      289 CONTINUE
C CALL SAVE(NM, 1, N, N, AA(1), 1, AA2)
C
      290 CONTINUE
C
SOLV 386
SOLV 387
SOLV 388
SOLV 389
SOLV 390
SOLV 391
SOLV 392
SOLV 393
SOLV 394
SOLV 395
SOLV 396
SOLV 397
SOLV 398
SOLV 399
SOLV 400
SOLV 401
SOLV 402
SOLV 403
SOLV 404
SOLV 405
SOLV 406
SOLV 407
SOLV 408
SOLV 409
SOLV 410
SOLV 411
SOLV 412
SOLV 413
SOLV 414
SOLV 415
SOLV 416
SOLV 417
SOLV 418
SOLV 419
SOLV 420

```

SOLVIT

SOLVIT

SOLV 421
SOLV 422

RETURN
END

SOLVIT

GETT

```
C      SUBROUTINE GETT(IU, IT, N1, A1, N2, A2)
C      DIMENSION A1(N1), A2(N2)
C      GO TO (10,20,30,40), IT
C      READ A1
C      10 READ(IU) A1
C         RETURN
C      READ N1 AND A1
C      20 READ(IU) N1, A1
C         RETURN
C      READ A1 AND A2
C      30 READ(IU) A1, A2
C         RETURN
C      READ IOUM AND A1
C      40 READ(IU) IOUM, A1
C         RETURN
C         END
```

```
GETT 001
GETT 002
GETT 003
GETT 004
GETT 005
GETT 006
GETT 007
GETT 008
GETT 009
GETT 010
GETT 011
GETT 012
GETT 013
GETT 014
GETT 015
GETT 016
GETT 017
GETT 018
GETT 019
GETT 020
GETT 021
GETT 022
```

GETT

SAVE

```
C      SUBROUTINE SAVE(IU, IT, N, N1, A1, N2, A2)
C      DIMENSION A1(N1), A2(N2)
C      GO TO (10,20,30,40), IT
C      WRITE A1
C      10 WRITE(IU) A1
C      RETURN
C      WRITE N AND A1
C      20 WRITE(IU) N, A1
C      RETURN
C      WRITE A1 AND A2
C      30 WRITE(IU) A1, A2
C      RETURN
C      WRITE N, A1, AND A2
C      40 WRITE(IU) N, A1, A2
C      RETURN
C      END
```

```
SAVE 001
SAVE 002
SAVE 003
SAVE 004
SAVE 005
SAVE 006
SAVE 007
SAVE 008
SAVE 009
SAVE 010
SAVE 011
SAVE 012
SAVE 013
SAVE 014
SAVE 015
SAVE 016
SAVE 017
SAVE 018
SAVE 019
SAVE 020
SAVE 021
SAVE 022
```

SAVE

ROWFM1

001 ROWF
 002 ROWF
 003 ROWF
 004 ROWF
 005 ROWF
 006 ROWF
 007 ROWF
 008 ROWF
 009 ROWF
 010 ROWF
 011 ROWF
 012 ROWF
 013 ROWF
 014 ROWF
 015 ROWF
 016 ROWF
 017 ROWF
 018 ROWF
 019 ROWF
 020 ROWF
 021 ROWF
 022 ROWF
 023 ROWF
 024 ROWF
 025 ROWF
 026 ROWF
 027 ROWF
 028 ROWF
 029 ROWF
 030 ROWF
 031 ROWF
 032 ROWF
 033 ROWF
 034 ROWF
 035 ROWF

SUBROUTINE ROWFM1(N, M, XI, YI, FI, XKD, NX, MX, LS, IF1)

C CALCULATES XKD ARRAY IN AUGMENTED FORM, ROW BY ROW.

C LS = 0, SURFACF SPLINE X + Y VARIABLE

C LS = 1, LINEAR SPLINE X CONSTANT

C LS = 2, LINEAR SPLINE Y CONSTANT

C DIMENSION XI(1), YI(1), XKD(1), FI(NX, MX)

C REWIND IF1

C N3 = N + 3

C N3M = N3 * M

C XKD(1) = 0.0

C XKD(2) = 0.0

C XKD(3) = 0.0

C FIRST ROW

C DO 10 J = 4, N3

C 10 XKD(J) = 1.0

C AUGMENT BY RHS

C J = N3

C DO 20 I = 1, M

C J = J + 1

C 20 XKD(J) = 0.0

C WRITE(IF1) (XKD(J), J = 1, N3M)

C SECOND ROW

C CHECK FOR LINEAR SPLINE X CONSTANT

C IF (LS .NE. 1) GO TO 25

C

ROWFM1

ROWF 036
 ROWF 037
 ROWF 038
 ROWF 039
 ROWF 040
 ROWF 041
 ROWF 042
 ROWF 043
 ROWF 044
 ROWF 045
 ROWF 046
 ROWF 047
 ROWF 048
 ROWF 049
 ROWF 050
 ROWF 051
 ROWF 052
 ROWF 053
 ROWF 054
 ROWF 055
 ROWF 056
 ROWF 057
 ROWF 058
 ROWF 059
 ROWF 060
 ROWF 061
 ROWF 062
 ROWF 063
 ROWF 064
 ROWF 065
 ROWF 066
 ROWF 067
 ROWF 068
 ROWF 069
 ROWF 070

```

C LINEAR SPLINE X CONSTANT
  XKD(2) = 1.0
  25 JX = 0
  DO 30 J = 4,N3
    JX = JX + 1
  30 XKD(J) = XI(JX)

C
C WRITE(IF1) (XKD(J), J = 1,N3M)

C
C THIRD ROW
  XKD(2) = 0.0
C CHECK FOR LINEAR SPLINE Y CONSTANT
  35 IF (LS .NE. 2) GO TO 37
  XKD(3) = 1.0
  37 JX = 0
  DO 40 J = 4,N3
    JX = JX + 1
  40 XKD(J) = YI(JX)

C
C WRITE(IF1) (XKD(J), J = 1,N3M)

C
C FOURTH THROUGH N3 ROWS
  DO 100 I = 1,N
    XKD(1) = 1.0
    XKD(2) = XI(I)
    XKD(3) = YI(I)
    J3 = 3
    DO 80 J = 1,N
      J3 = J3 + 1
      AIJ = 0.0
      TRI = 0.0
      TR2 = 0.0
      IF (I.EQ. J) GO TO 70
      IF (XI(I).EQ. XI(J)) GO TO 50
      TRI = (XI(I) - XI(J))**2
  70
  80
  100
  
```

ROWFM1

ROWF 071
ROWF 072
ROWF 073
ROWF 074
ROWF 075
ROWF 076
ROWF 077
ROWF 078
ROWF 079
ROWF 080
ROWF 081
ROWF 082
ROWF 083
ROWF 084
ROWF 085
ROWF 086
ROWF 087
ROWF 088
ROWF 089
ROWF 090

```
50 IF (YI(I) .EQ. YI(J)) GO TO 60
   TR2 = (YI(I) - YI(J))*2
60 RAIJ = TR1 + TR2
   IF (RAIJ .EQ. 0) GO TO 70
   AIJ = RAIJ*ALOG(RAIJ)
70 XKD(J3) = AIJ
   A0 CONTINUE
C
C AUGMENT THE ROW
   DC - J = 1, M
   J3 = J3 + 1
90 XKD(J3) = +FI(I, J)
   WRITE(IFI) (XKD(J), J = 1, N3M)
C
C 100 CONTINUE
C
C RETURN
   END
```

ROWFM1

FLOW

001C
 FLOW 002C
 FLOW 0031
 FLOW 004
 FLOW 005
 FLOW 006
 FLOW 007
 FLOW 008
 FLOW 009
 FLOW 010
 FLOW 011
 FLOW 012
 FLOW 013
 FLOW 014
 FLOW 015
 FLOW 016
 FLOW 017
 FLOW 018
 FLOW 019
 FLOW 020
 FLOW 021
 FLOW 022
 FLOW 023
 FLOW 024
 FLOW 025
 FLOW 026
 FLOW 027
 FLOW 028
 FLOW 029
 FLOW 030
 FLOW 031
 FLOW 032
 FLOW 033
 FLOW 034
 FLOW 035

```

OVERLAY (MARK4,2,1)
PROGRAM FLOW
SURROUTINE FLOW

C THIS IS THE EXECUTIVE CONTROL PROGRAM FOR THE FLOW FIELD ANALYSIS
C ROUTINES

COMMON /EXEC/CASE,ITITL,PAGE,ERROR
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,(APFI,TAPEJ,TAPEK
COMMON /TAGS/ITAG4,ITAG9,ITAG10
COMMON /FIELD/TITLEM(10),TTITLES(10),TTITLEA(10),TTITLER(10),
1 IMYAR(9),IDTYP(5),LOSFT(5),LORAR(20),L,
2 DINR(6),IRW,LCNEXT,IREG,MACH,NAB,ALPHA,BFTA,NREG,
3 NSREG,ITFLAG
COMMON /FLIGHT/EACH,ALT,POSTAG,TSTAG,V,REND,PFS,TFS,AFS,RHOF,S,VIS
COMMON /FSRS/FS(8),BS(8)

C DIMENSION TITLE(15),F(12),F2(13),F3(17)
C
PEAL MACH
INTEGER ERROR,PAGE
INTEGER TAPEIN,TAPEUT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK

C WRITE (TAPEOT,5)
C 5 FORMAT (I11,23H** FLOW=FIELD OPTION ** )
C MASTER DIRECTORY INFORMATION
C READ (TAPEIN,10) MFLAG,TITLEM
C 10 FORMAT (I11,19X,10A4)
C
  
```

FLOW

FLOW

```

C CHECK UNIT STATUS, MFLAG = 0 SET UP COMPLETE NEW UNIT, MFLAG = 1
C USE OLD UNIT POINTERS,
  IG10 = 1
  IF (MFLAG .NE. 0) GO TO 30
C INITIALIZE MASTER DIRECTORY TABLE ARRAYS, ETC.
C MASTER DIRECTORY NAME
  CALL WRITMS (10, TITL, 10, IG10)
C WRITE (10=IG10) TITL
  DO 20 I=1, 9
20  IMTAB(I) = 0
  IMTAR(1) = 0
  IMTAR(2) = 4
  IMTAR(5) = 4
  LCNEXT = 4
  LOSET(1) = 4
  GO TO 33
C
C OLD DATA UNIT, READ POINTERS FROM MASTER DIRECTORY TABLE (IMTAB)
30 CONTINUE
  CALL READMS(10, TITL, 10, IG10)
  READ(10=IG10) TITL
  IG10 = 2
  CALL READMS (10, IMTAR, 9, IG10)
  READ (10=IG10) IMTAB
  LCNEXT = IMTAB(2)
C
33 DO 35 I=1, 5
35 LOSET(I) = IMTAB(I+4)
  NDSET = IMTAR(1)
C
C SET DIRECTORY
C

```

```

FLOW 036
FLOW 037
FLOW 038
FLOW 039
FLOW 040
FLOW 041
FLOW 042
FLOW 043C
FLOW 044I
FLOW 045
FLOW 046
FLOW 047
FLOW 048
FLOW 049
FLOW 050
FLOW 051
FLOW 052
FLOW 053
FLOW 054
FLOW 055
FLOW 056
FLOW 057C
FLOW 058I
FLOW 059
FLOW 060C
FLOW 061I
FLOW 062
FLOW 063
FLOW 064
FLOW 065
FLOW 066
FLOW 067
FLOW 068
FLOW 069
FLOW 070

```

FLOW

FLOW

FLOW 071
 FLOW 072
 FLOW 073
 FLOW 074
 FLOW 075
 FLOW 076
 FLOW 077
 FLOW 078
 FLOW 079
 FLOW 080
 FLOW 081
 FLOW 082
 FLOW 083
 FLOW 084
 FLOW 085
 FLOW 086
 FLOW 087
 FLOW 088
 FLOW 089
 FLOW 090C
 FLOW 091I
 FLOW 092
 FLOW 093
 FLOW 094
 FLOW 095
 FLOW 096
 FLOW 097C
 FLOW 098I
 FLOW 099
 FLOW 100
 FLOW 101
 FLOW 102
 FLOW 103
 FLOW 104
 FLOW 105

```

C INPUT LAST SET FLAG (LASTS), NEW SET FLAG (NEWS), DATA SET NUMBER
C (NSFT), MACH NUMBER, SET TITLE (TTITLES)
40 READ (TAPEIN,50) LASTS,NEWS,NSFT,MACH,TTITLES
50 FORMAT (2I1,IX,I1,F6.0,10X,10A4)

C
C CHECK DATA SET STATUS,
C IF (NEWS .NE. 0) GO TO 70
C
60 DO 60 I=1,20
   LOAB(I) = 0
   NDSET = 1
   IMTAB(5) = LCNFXT
   LOSET(I) = LCNEXT
   LCNEXT = LCNFXT + 3
   GO TO 90

C
C READ POINTERS OF EXISTING SET TO GET TO LOWER DIRECTORY TABLES,
C 70 IF (NEWS .NE. 1) GO TO 80
   IG10 = IMTAB(NSSET+4)
   CALL READMS (10,E,12,IG10)
   READ (10,IG10) F
   DO 71 J=1,10
     71 TTITLES(I) = E(I)
     MACH = E(11)
     NAR = F(12)
     IG10 = IG10 + 1
   CALL READMS (10,LOAB,20,IG10)
   READ (10,IG10) LOAB
   GO TO 90

C
C ADD NEW SET TABLE DATA TO OLD
A0 NDSET = NDSET + 1
   DO 81 I=1,20
     A1 LOAB(I) = 0
     LCNEXT(NSSET) = LCNEXT
  
```

FLOW

FLOW

```

C      IMTAB(NSET+4) = LCNEXT
C      LCNEXT = LCNEXT + 3
C
C      INPUT LAST ALPHA-BETA FLAG (LASTAR), NEW ALPHA-BETA SET FLAG
C      (NEWAR), ALPHA-BETA SET NUMRER (IAB), ALPHA-BETA VALUES,
C      ALPHA-BETA SET TITLE (TITLEA),
C      90 READ (TAPEIN,100) LASTAB,NFWAB,IAR,ALPHA,BETA,TITLEA
C      100 FORMAT (2I1,7F2,2F6.0,4X,10A4)
C
C      CHECK ALPHA-BETA SET STATUS
C      IF (NEWAR .GT. 0) GO TO 120
C
C      INITIALIZE ALPHA-BETA DIRECTORY
C      DO 110 I=1,20
C      110 LORG(I) = 0
C
C      NAR = 1
C      LOAR(IAB) = LCNEXT
C      LCNEXT = LCNEXT + 3
C      GO TO 140
C
C      READ POINTERS OF EXISTING ALPHA-BETA DIRECTORY
C      120 IF (NEWAB .NE. 1) GO TO 130
C      IG10 = LOAB(IAB)
C      CALL READMS (10,E2,13,IG10)
C      READ (10,IG10) E2
C      DO 121 I=1,10
C      121 TITLEA(I) = E2(I)
C      ALPHA = E2(11)
C      BETA = E2(12)
C      NREG = E2(13)
C      IG10 = IG10 + 1
C      CALL READMS (10,LOG,20,IG10)

```

FLOW 106
FLOW 107
FLOW 108
FLOW 109
FLOW 110
FLOW 111
FLOW 112
FLOW 113
FLOW 114
FLOW 115
FLOW 116
FLOW 117
FLOW 118
FLOW 119
FLOW 120
FLOW 121
FLOW 122
FLOW 123
FLOW 124
FLOW 125
FLOW 126
FLOW 127
FLOW 128
FLOW 129
FLOW 130
FLOW 131
FLOW 132C
FLOW 133I
FLOW 134
FLOW 135
FLOW 136
FLOW 137
FLOW 138
FLOW 139
FLOW 140C

FLOW

FLOW

FLOW 141X
 FLOW 142
 FLOW 143
 FLOW 144
 FLOW 145
 FLOW 146
 FLOW 147
 FLOW 148
 FLOW 149
 FLOW 150
 FLOW 151
 FLOW 152
 FLOW 153
 FLOW 154
 FLOW 155
 FLOW 156
 FLOW 157
 FLOW 158
 FLOW 159
 FLOW 160
 FLOW 161
 FLOW 162
 FLOW 163
 FLOW 164
 FLOW 165
 FLOW 166
 FLOW 167
 FLOW 168
 FLOW 169
 FLOW 170
 FLOW 171
 FLOW 172
 FLOW 173
 FLOW 174C
 FLOW 175I

```

C      READ (10,IG10) LORG
C      GO TO 140
C
C      ADD NEW ALPHA-BETA DIRECTORY
130   NAR = NAB + 1
      DO 131 I=1,20
131   LORG(I) = 0
      LOAR(IAB) = LCNEXT
      LCNEXT = LCNEXT + 3
C
140   DO 141 I=1,5
141   IDTYP(I) = 0
C
C      INPUT LAST REGION GROUP FLAG (LASTR), NEW REGION SET FLAG
C      (NEWR), REGION SET NUMBER (IRFG), DATA TYPE FLAGS (IDTYP),
C      REGION TITLE (TITLER).
      READ (TAPEIN,150) LASTR,NEWR,IREG, IDTYP(1), ISORCE,IRW,TITLER
150   FORMAT (2I1,12,3I1,13X,10A4)
C      CHECK REGION SET STATUS
      IF (NEWR .NE. 0) GO TO 170
C
C      INITIALIZE FLOW FIELD DATA
      NREG = 1
      NSREG = 0
      ITFLAG = 0
      IG10 = LCNEXT
      DO 161 I=1,10
161   E3(I) = TITLER(I)
      DO 162 I=1,5
162   E3(I+10) = IDTYP(I)
      E3(16) = NSRFG
      E3(17) = ITFLAG
      CALL WRITMS (10,E3,17,IG10)
C      WRITE (10,IG10) E3
  
```

FLOW

FLOW

FLOW 176
 FLOW 177
 FLOW 178
 FLOW 179
 FLOW 180
 FLOW 181
 FLOW 182
 FLOW 183C
 FLOW 184I
 FLOW 185
 FLOW 186
 FLOW 187
 FLOW 188
 FLOW 189
 FLOW 190
 FLOW 191
 FLOW 192
 FLOW 193
 FLOW 194
 FLOW 195
 FLOW 196
 FLOW 197
 FLOW 198
 FLOW 199
 FLOW 200
 FLOW 201
 FLOW 202
 FLOW 203C
 FLOW 204I
 FLOW 205
 FLOW 206
 FLOW 207
 FLOW 208
 FLOW 209
 FLOW 210

FLOW

```

    LOG(IREG) = IG10
    LCNEXT = IG10 + 5
    GO TO 190

C   READ POINTERS OF EXISTING FLOW FIELD DATA SET
170 IF (NEW, NE, 1) GO TO 180
    IG10 = LOG(IREG)
    CALL P=ADMS (10,E3,17,IG10)
    READ (10,IG10) E3
    DO 171 I=1,10
171  TITLER(I) = F3(I)
    DO 172 I=1,5
172  IDTYP(I) = E3(I+10)
    NSREG = E3(16)
    ITFLAG = E3(17)
    GO TO 190

C   ADD NEW FLOW FIELD TABLF ITEM
180 NSREG = 0
    ITFLAG = 0
    IG10 = LCNEXT
    DO 182 I=1,10
182  E3(I) = TITLER(I)
    DO 183 I=1,5
183  F3(I+10) = IDTYP(I)
    E3(16) = NSREG
    E3(17) = ITFLAG
    CALL WRITMS (10,E3,17,IG10)
    WRITE (10,IG10) E3
    NREG = NREG + 1
    LOG(IREG) = IG10
    LCNEXT = IG10 + 5

C
C   190 CONTINUE
  
```

FLOW

```

C SELECT TYPE OF DATA TO BE READ IN OR GENERATED
C IDTYP(1) = 1 (FLOW FIELDS), #2 (SURFACE DATA),
C # 3 (STREAMLINES), #4 (SHOCK SHAPE)
IF (IDTYP(1).GT.0 .AND. IDTYP(1).LE.4) GO TO 200
WRITE (TAPEOT,202) IDTYP(1)
202 FORMAT (I4,14H**IDTYP(1) INPUT =,I5,I7H IS OUT OF RANGE.,
1 14H PROGRAM STOP. )
STOP
C
200 ITAB = IDTYP(1)
GO TO (210,220,210,240), ITAB
C FLOW FIELD DATA
210 IF (ISORCF.GT.0 .AND. ISORCF.LE.4) GO TO 212
WRITE (TAPEOT,211) ISORCF
211 FORMAT (I4,14H**ISORCF INPUT =,I5,I7H IS OUT OF RANGE.,
1 14H PROGRAM STOP. )
STOP
212 ITAR = ISORCF
GO TO (213,214,215,216), ITAR
C
C HAND INPUT DATA
213 CALL FFINPT
GO TO 600
C
C SHOCK-EXPANSION GENERATED DATA
214 CALL FFBODY
GO TO 600
C
C FUTURE ADDITION
215 CALL FFAIRF
GO TO 600
C
C SIMPLE FLOW FIELDS
216 CALL FFSPEC
GO TO 600

```

FLOW 211
FLOW 212
FLOW 213
FLOW 214
FLOW 215
FLOW 216
FLOW 217
FLOW 218
FLOW 219
FLOW 220
FLOW 221
FLOW 222
FLOW 223
FLOW 224
FLOW 225
FLOW 226
FLOW 227
FLOW 228
FLOW 229
FLOW 230
FLOW 231
FLOW 232
FLOW 233
FLOW 234
FLOW 235
FLOW 236
FLOW 237
FLOW 238
FLOW 239
FLOW 240
FLOW 241
FLOW 242
FLOW 243
FLOW 244
FLOW 245

FLOW

FLOW

FLOW 246
 FLOW 247
 FLOW 248
 FLOW 249
 FLOW 250
 FLOW 251
 FLOW 252
 FLOW 253
 FLOW 254
 FLOW 255
 FLOW 256
 FLOW 257
 FLOW 258
 FLOW 259
 FLOW 260
 FLOW 261
 FLOW 262
 FLOW 263
 FLOW 264
 FLOW 265
 FLOW 266
 FLOW 267
 FLOW 268C
 FLOW 269I
 FLOW 270
 FLOW 271
 FLOW 272
 FLOW 273
 FLOW 274
 FLOW 275
 FLOW 276
 FLOW 277
 FLOW 278
 FLOW 279
 FLOW 280

FLOW

```

C SURFACE VELOCITY VECTOR DATA
C 220 CALL FFSURF
      GO TO 600
C
C SHOCK SHAPE DATA (NOT ACTIVE NOW)
C 240 CONTINUE
      GO TO 600
C
C 600 CONTINUE
      IF (IRW .NE. 0) GO TO 612
C COMPLETE FLOW FIELD DATA TABLE
      IG10 = LORG(IREG)
      DO 601 I=1,10
        601 E3(I) = TITLER(I)
      DO 602 I=1,5
        602 F3(I+10) = IDTYP(I)
      E3(16) = NSREG
      E3(17) = ITFLAG
      CALL WPITMS (10,E3,17,IG10)
C WRITE (10,IG10) E3
      WRITE (TAPEOT,610) TITLER,IDTYP,NSREG,ITFLAG
        610 FORMAT (1H0,5X,10H** TITLER=,10A0,5X,6HIDTYP=,5I2,3X,6HNSREG=,
          1 I3,3X,7HITFLAG=,I3)
C
C CHECK FOR LAST REGION
C 612 IF (LASTR .EQ. 0) GO TO 140
      IF (IRW .NE. 0) GO TO 622
C COMPLETE REGION DIRECTORY
      IG10 = LOAB(IAB)
      DO 613 I=1,10
  
```

FLOW

```

613 E2(I) = TITLFA(I)
    E2(11) = ALPHA
    E2(12) = BETA
    E2(13) = NREG
    CALL WRITMS (10,E2,13,IG10)
    WRITE (10,IG10) E2
    WRITE (TAPE07,620) TITLEA,ALPHA,BETA,NREG,LORG
620 FORMAT (1H,10H** TITLFA=,10A4,7H ALPHA=,F6.3,6H BETA=,F6.3,
    1 8H NREG=,I4,1H ,8H** LORG=,20I4)
    IG10 = IG10 + 1
    CALL WRITMS (10,LORG,20,IG10)
    WRITE (10,IG10) LORG
C
C
C
C
C CHECK FOR LAST ALPHA-BETA
622 IF (LASTAB .EQ. 0) GO TO 90
    IF (TRM .NE. 0) GO TO 632
C
C COMPLETF SET DIRECTORY
    IG10 = LDSET(MSET)
    DO 623 I=1,10
623 E(I) = TITLES(I)
    E(11) = MACH
    F(12) = NAB
    CALL WRITMS (10,F,12,IG10)
    WRITE (10,IG10) F
    WRITE (TAPE07,630) TITLES,MACH,NAB,LOAD
630 FORMAT (1H,10H** TITLFS=,10A4,6H MACH=,F6.3,5H NAB=,I4,1H ,
    1 8H** LOAD=,20I4)
    IG10 = IG10 + 1
    CALL WRITMS (10,LOAD,20,IG10)
    WRITE (10,IG10) LOAD
C
C CHECK FOR LAST MASTER SET

```

```

FLOW 281
FLOW 282
FLOW 283
FLOW 284
FLOW 285C
FLOW 286I
FLOW 287
FLOW 288
FLOW 289
FLOW 290
FLOW 291C
FLOW 292I
FLOW 293
FLOW 294
FLOW 295
FLOW 296
FLOW 297
FLOW 298
FLOW 299
FLOW 300
FLOW 301
FLOW 302
FLOW 303
FLOW 304
FLOW 305
FLOW 306C
FLOW 307I
FLOW 308
FLOW 309
FLOW 310
FLOW 311
FLOW 312C
FLOW 313I
FLOW 314
FLOW 315

```

FLOW

FLOW

FLOW 316
FLOW 317
FLOW 318
FLOW 319
FLOW 320
FLOW 321C
FLOW 322I
FLOW 323
FLOW 324
FLOW 325
FLOW 326
FLOW 327I
FLOW 328C
FLOW 329

```
632 IF (LASTS .EQ. 0) GO TO 40
    IF (IRM .NE. 0) GO TO 650
    IMTAB(1) = NDSET
    IMTAB(2) = LCNEXT
    IG10 = 2
    CALL WRITMS (10,IMTAB,9,IG10)
    WRITE (10:IC10) IMTAB
    WRITE (TAPEUT,640) IMTAB
    640 FORMAT (1H ,A#*IMTAB#,9I4)
C
C
C650 RETURN
    650 CONTINUE
    END
```

FLOW

MERID

```

MERI 001
MERI 002
MERI 003
MERI 004
MERI 005
MERI 006
MERI 007
MERI 008
MERI 009
MERI 010
MERI 011
MERI 012
MERI 013
MERI 014
MERI 015
MERI 016
MERI 017
MERI 018
MERI 019
MERI 020
MERI 021
MERI 022
MERI 023
MERI 024
MERI 025
MERI 026
MERI 027
MERI 028
MERI 029
MERI 030
MERI 031
MERI 032
MERI 033
MERI 034
MERI 035

```

```

SURROUTINE MERID
C
C GENERALIZED CUTTING PLANE OF MERIDIAN TYPE
C -- THAT IS, CUTTING PLANE IS PARALLEL TO AXIS
C
C
C COMMON /EXFC/ CASE, TITLE(15), PAGE, ERROR
COMMON /TAPE/TAPFIN,TAPEOT,TAPFA,TAPEB,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
C DIMENSION ELEM(25), XN(4), YN(4), ZN(4), YP(4), ZP(4), PHI(37),
1 A(4), MP(4), MPI(10), MPI(10), NINT(40), XREC(6), XP(4),
2 XI(10,2), YI(10,2), ZI(10,2), RI(10,2), XA(10,2),
3 COMPID(10), NT(10), NI(10)
4 ,YPA(40), F(25), EP(25), IPANL(10)
C
C INTEGER FPROP,PAGE,NREC(40,50),SYMFACT
INTEGER TAPFIN,TAPEOT,TAPE,TAPEB,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C REAL LR
C
C EQUIVALENCE (ELEM(11), XN(1)), (ELEM(15), YN(1)),
1 (ELEM(19), ZN(1))
C DATA PI,RC/3.14159265,1.74532925E-2/
C
C READ IN PANEL ID AND CUTTING PLANE CONTROL CARD
C READ (TAPEIN,7) IPANL,NPL,NPHI,ISYM
7 FORMAT (10I2,12,2I1)
C IPRINT = 1
C IPRINT = 1
C REMIND TAPEB

```

MERID

MERID

MERI 036
 MERI 037
 MERI 038
 MERI 039
 MERI 040
 MERI 041
 MERI 042
 MERI 043
 MERI 044
 MERI 045
 MERI 046
 MERI 047
 MERI 048
 MERI 049
 MERI 050
 MERI 051
 MERI 052
 MERI 053
 MERI 054
 MERI 055
 MERI 056
 MERI 057
 MERI 058
 MERI 059
 MERI 060
 MERI 061
 MERI 062
 MERI 063
 MERI 064
 MERI 065
 MERI 066
 MERI 067
 MERI 068
 MERI 069
 MERI 070

MERID

```

C      REMIND TAPEC
C
C      READ IN CUTTING PLANE ORIGIN AND ORIENTATION
C
C      READ (TAPEIN,30) XPO,YPO,ZPO,PSIO,THEO,PHIO
C      30 FORMAT(6F10.0)
C      CONSTANTS RELATING TO CUTTING PLANE AXIS
C      SINDS = SIN(RC*PSIO)
C      COSPS = COS(RC*PSIO)
C      COST = COS(RC*THEO)
C      SINT = SIN(RC*THEO)
C      STSPS = SINDS*SINT
C      STCPS = COSPS*SINT
C      CTSPS = COST*SINDS
C
C      IF (NPL.GT. 36) NPL = 36
C      IF (INPHI .LT. 2) GO TO 130
C
C      PARALLEL CUTTING PLANES. READ IN CONSTANT PHI VALUE.
C      READ (TAPEIN,50) PHICD
C      50 FORMAT(F10.6)
C      PHI(1) = RC*PHICD
C      TAMP = TAN(PHI(1))
C
C      IF (INPHI .EQ. 3) GO TO 90
C      EQUAL SPACING, READ IN END VALUES.
C      READ (TAPEIN,60) (XN(I),YN(I),ZN(I),I=1,2)
C      60 FORMAT(3F10.0)
C
C      FOR EQUAL SPACING, MUST HAVE AT LEAST TWO PLANES
C      IF (NPL .LT. 2) NPL = 2

```

MERID

```

C PROJECT INTO VIEWING PLANE AND FIND AXIS LOCATIONS.
C DO 70 N = 1,2
  XX = XN(N) - XPO
  YY = YN(N) - YPO
  ZZ = ZN(N) - ZPO
  VP(N) = -XX*SINPS + YY*COSPS
  ZP(N) = XX*STCPS + YY*STSPS + Z7*CUST
  70 YPA(N) = YP(N) + ZP(N)*TAMP
  DYPA = (YPA(2) - YPA(1))/(NPL - 1)
C
  NINT(1) = 0
  NPL1 = NPL
  DO 80 N = 2,NPL1
    PHI(N) = PHI(1)
  NINT(N) = 0
  80 YPA(N) = YPA(N-1) + DYPA
  GO TO 110

```

```

C VARIABLE PLANE SPACING, READ IN POSITIONS,
C PROJECT INTO VIEWING PLANE AND FIND AXIS LOCATIONS.
C DO NPL1 = NPL
  DO 100 N = 1,NPL
    READ (TAPEIN,60) XN(1),YN(1),ZN(1)
    XX = XN(1) - XPO
    YY = YN(1) - YPO
    ZZ = ZN(1) - ZPO
    VP(1) = -XX*SINPS + YY*COSPS
    ZP(1) = XX*STCPS + YY*STSPS + ZZ*CUST
    YPA(N) = YP(1) + ZP(1)*TAMP
    PHI(N) = PHI(1)
  NINT(N) = 0
  100 CONTINUE

```

MERI 071
 MERI 072
 MERI 073
 MERI 074
 MERI 075
 MERI 076
 MERI 077
 MERI 078
 MERI 079
 MERI 080
 MERI 081
 MERI 082
 MERI 083
 MERI 084
 MERI 085
 MERI 086
 MERI 087
 MERI 088
 MERI 089
 MERI 090
 MERI 091
 MERI 092
 MERI 093
 MERI 094
 MERI 095
 MERI 096
 MERI 097
 MERI 098
 MERI 099
 MERI 100
 MERI 101
 MERI 102
 MERI 103
 MERI 104
 MERI 105

MERID

MERID

MERI 106
 MERI 107
 MERI 108
 MERI 109
 MERI 110
 MERI 111
 MERI 112
 MERI 113
 MERI 114
 MERI 115
 MERI 116
 MERI 117
 MERI 118
 MERI 119
 MERI 120
 MERI 121
 MERI 122
 MERI 123
 MERI 124
 MERI 125
 MERI 126
 MERI 127
 MERI 128
 MERI 129
 MERI 130
 MERI 131
 MERI 132
 MERI 133
 MERI 134
 MERI 135
 MERI 136
 MERI 137
 MERI 138
 MERI 139
 MERI 140

```

110 IF (IPRNT.EQ. 1)
111   IWRITE (TAPEOT,120) NPL,PHICD,(YPA(N),N=1,NPL)
112   FORMAT(1H1, I3, 32H PARALLFL CUTTING PLANES PHI = , F10.6/
113     1 140, (6F20.5))
114   GO TO 180
115
116 CONTINUE
117 IF (IMPHI.EQ. 0) GO TO 160
118 READ (TAPEIN,140) (PHI(I),I=1,NPL)
119   FORMAT(6F10.6)
120   DPHI = PHID*RC
121   DO 150 I = 1,NPL
122     NINT(I) = 0
123     YPA(I) = 0.0
124     PH7(I) = PHI(I)*RC + DPHI
125     NPLI = NPL
126   GO TO 180
127 CONTINUE
128 NPLI = NPL + 1
129   DPHI = 360./(NPL) *RC
130   PH7(1) = PHID*RC
131   NINT(1) = 0
132   YPA(1) = 0.0
133   DO 170 I = 2,NPLI
134     NINT(I) = 0
135     YPA(I) = 0.0
136     PH7(I) = PHI(I-1) + DPHI
137
138   PH7(NPLI) = PH7(1)
139   MPTOT = 0
140   DO 190 I = 1,10
141     NT(I) = 0
142
143

```

MERID

MERID

MERI	141
MERI	142
MERI	143I
MERI	144C
MERI	145
MERI	146
MERI	147
MERI	148
MERI	149
MERI	150
MERI	151
MERI	152
MERI	153
MERI	154
MERI	155I
MERI	156C
MERI	157
MERI	158
MERI	159
MERI	160
MERI	161
MERI	162
MERI	163
MERI	164
MERI	165
MERI	166
MERI	167I
MERI	168C
MERI	169
MERI	170
MERI	171
MERI	172
MERI	173
MERI	174
MERI	175

MERID

```

C RETRIEVE ELEMENT DATA FROM STORAGE
  IGA = 2
  READ (4*IG4) E
  CALL READMS (4,E,25,IG4)
  NPF = E(2)
  NREM = E(4)

C
C
C START OF PANEL CYCLE
  DO 675 JJ=1,10
  NT(JJ) = MPTOT + 1
  IF (IPANL(JJ) .EQ. 0) GO TO 680
  IF (IPANL(JJ) .GT. NPE) GO TO 600
  IGA = IPANL(JJ)*5
  READ (4*IG4) EP
  CALL READMS (4,EP,25,IG4)
  COMID = EP(2)
  ISTART = EP(3)
  LL = EP(4)
  SYMFC = EP(A)
  COMPID(JJ) = COMID
  ISTART = ISTART + NREM

C
C START OF ELEMENT DO LOOP
  DO 670 J=1,LL
  IGA = ISTART + NREM*J
  READ (4*IG4) ELEM
  CALL READMS (4,ELEM,25,IG4)
  JE = J

C
C PROJECT FOUR CORNER POINTS INTO T2,T3 PLANE
  (IN COORDS RELATIVE TO CUTTING PLANE AXYS),
C FIRST CHECK IF DUMMY, ZERO AREA ELEMENT - IF SO, SKIP OVER.
  IF (ELEM(10) .LE. 1.0E-5) GO TO 670
C

```

MERID

```
C ALWAYS PROJECT LINES 2-3 AND 3-4
  NIT= 2
  NR = 2
  NF = 4
  ICOL = ELEM(2) + 0.01
  IROW = ELEM(3) + 0.01
C CHECK IF NEW ROW
  IF (IROW .NE. 1) GO TO 250
  NR = 1
  NF = 1
C CHECK IF NEW COLUMN
  250 IF (ICOL .NE. 1) GO TO 260
  NIT= 1
  NR = 1
  260 CONTINUE
C
C PRESET NPI ARRAY
  DO 270 K = 1,10
  270 NPI(K) = 0
C
  IO = 0
  DO 340 N = NR,4
  XX = XN(N) = XPO
  YY = YN(N) = YPO
  ZZ = ZN(N) = ZPO
C
  XP(N) = XX*CTCPS + YY*CYSPS - ZZ*SZNT
  YP(N) = -YX*SIKPS + YY*CSKPS
  ZP(N) = XX*STCPS + YY*STSPS + ZZ*CSST
C
  MP(N) = 0
  IF (INPHY .GT. 1) GO TO 290
  IF ((ZP(N).EQ.0.0) .AND. (YP(N).EQ.0.0)) GO TO 350
  IF (JE .GT. LL) GO TO 280
  IF ((ZP(N).LT.0.0) .AND. (YP(N).EQ.0.0)) GO TO 360
```

MERI 176
MERI 177
MERI 178
MERI 179
MERI 180
MERI 181
MERI 182
MERI 183
MERI 184
MERI 185
MERI 186
MERI 187
MERI 188
MERI 189
MERI 190
MERI 191
MERI 192
MERI 193
MERI 194
MERI 195
MERI 196
MERI 197
MERI 198
MERI 199
MERI 200
MERI 201
MERI 202
MERI 203
MERI 204
MERI 205
MERI 206
MERI 207
MERI 208
MERI 209
MERI 210

MERID

MERID

MERI 211
 MERI 212
 MERI 213
 MERI 214
 MERI 215
 MERI 216
 MERI 217
 MERI 218
 MERI 219
 MERI 220
 MERI 221
 MERI 222
 MERI 223
 MERI 224
 MERI 225
 MERI 226
 MERI 227
 MERI 228
 MERI 229
 MERI 230
 MERI 231
 MERI 232
 MERI 233
 MERI 234
 MERI 235
 MERI 236
 MERI 237
 MERI 238
 MERI 239
 MERI 240
 MERI 241
 MERI 242
 MERI 243
 MERI 244
 MERI 245

MERID

```

280 CONTINUE
C
  A(N) = ATAN2(+YP(N),+ZP(N))
  IF (A(N) .LE. 0.0) A(N) = 2.*PI + A(N)
  IF (INPHI .NE. 0) GO TO 320
  MP(N) = A(N)/DPHI + 1
  GO TO 360

C
  MP CALCULATIONS FOR PARALLEL PLANES
290 CONTINUE
  A(N) = YP(N) + ZP(N)*TANP
  IF (INPHI .EQ. 3) GO TO 300
  PM = (A(N) - YPA(1))/DYPA
  IF (PM .LE. 0.0) GO TO 360
  MP(N) = PM + 1
  IF (MP(N) .GT. NPL) MP(N) = NPL
  GO TO 360

C
300 CONTINUE
  DO 310 K = 1,NPL1
  IP = K
  IF (YPA(K) .GT. A(N)) GO TO 340
310 CONTINUE
  MP(N) = IP
  GO TO 360

C
  C
320 DO 330 K = 1,NPL1
  IP = K
  IF (PHI(K) .GT. A(N)) GO TO 340
330 CONTINUE
  IF (JE .GT. 1L) GO TO 360
  IP = IP + 1
340 MP(N) = IP - 1
  GO TO 360
  
```

MERID

MERI 246
MERE 247
MERE 248
MERE 249
MERE 250
MERE 251
MERE 252
MERE 253
MERE 254
MERE 255
MERE 256
MERE 257
MERE 258
MERE 259
MERE 260
MERE 261
MERE 262
MERE 263
MERE 264
MERE 265
MERE 266
MERE 267
MERE 268
MERE 269
MERE 270
MERE 271
MERE 272
MERE 273
MERE 274
MERE 275
MERE 276
MERE 277
MERE 278
MERE 279
MERE 280

```
350 ID = IO + 1
360 CONTINUE
C CHECK FOR CONTINUITY IN ANGLE (REFLECTED SYMMETRIC ELEMENTS ONLY).
  IF (JE .LE. LL) GO TO 380
  DO 370 N = NR, 4
  IF (MP(N) .EQ. 1) MP(N) = NPL1
370 CONTINUE
380 CONTINUE
C CHECK FOR ORIGIN POINTS (ASSUMED IN PAIRS)
  IF (TO .EQ. 0) GO TO 420
  N1 = 1
  N2 = 2
  N3 = 3
  N4 = 4
390 IF (MP(N1) .NE. 0) GO TO 410
  IF (MP(N2) .EQ. 0) GO TO 400
C
  MP(N1) = MP(N2)
  MP(N4) = MP(N3)
  GO TO 420
C
  400 MP(N1) = MP(N4)
  MP(N2) = MP(N3)
  GO TO 420
C
  410 NN = N1
  N1 = N2
  N2 = N3
  N3 = N4
  N4 = NN
  IF (N4 .NE. 4) GO TO 390
C
  420 CONTINUE
```

MERID

MERID

MERI 281
 MERI 282
 MERI 283
 MERI 284
 MERI 285
 MERI 286
 MERI 287
 MERI 288
 MERI 289
 MERI 290
 MERI 291
 MERI 292
 MERI 293
 MERI 294
 MERI 295
 MERI 296
 MERI 297
 MERI 298
 MERI 299
 MERI 300
 MERI 301
 MERI 302
 MERI 303
 MERI 304
 MERI 305
 MERI 306
 MERI 307
 MERI 308
 MERI 309
 MERI 310
 MERI 311
 MERI 312
 MERI 313
 MERI 314
 MERI 315

```

C
C
C YFST FOR INTERSECTIONS AND FIND THEM IF INDICATED
  MPIN = 1
  NIMP = 1
  ICT = +1
  MPMAX = 0
  IPI = NII
C
C 430 IPP = IPI + 1
  IF (IP2 .GT. 4) IP2 = 1
  IF (MP(IP2) = MP(IPI)) 460, 440, 450
C
C CHECK FOR LAST CORNER POINT
C 440 IF (IP2 .EQ. NF) GO TO 600
C
C GO TO NEXT CORNER POINT
  IPI = IP2
  GO TO 430
C
C
C INTERSECTION OF NMP = MP(IPI) + 1 WITH LINE SEGMENT
C 450 NMP = MP(IPI) + 1
  IMP = +1
  GO TO 470
C
C
C INTERSECTION OF NMP = MP(IPI) WITH LINE SEGMENT
C 460 NMP = MP(IPI)
  IMP = -1
C
C
C CHECK FOR FIRST INTERSECTION
C 470 IF (MPMAX .EQ. 0) GO TO 540

```

MERID

MERID

MERI	316
MERI	317
MERI	318
MERI	319
MERI	320
MERI	321
MERI	322
MERI	323
MERI	324
MERI	325
MERI	326
MERI	327
MERI	328
MERI	329
MERI	330
MERI	331
MERI	332
MERI	333
MERI	334
MERI	335
MERI	336
MERI	337
MERI	338
MERI	339
MERI	340
MERI	341
MERI	342
MERI	343
MERI	344
MERI	345
MERI	346
MERI	347
MERI	348
MERI	349
MERI	350

```

C C PREVIOUS INTERSECTIONS, CHECK FOR TURNING CORNER
C C AND COMING BACK#,
C C IF (IMP)480,480,500
C C 480 IF (MP(IP1) = MP(IP1=1))530,490,520
C C 490 IF (MP(IP1=1) = MP(IP1=2))530,520,520
C C 500 IF (MP(IP1) = MP(IP1=1))520,510,570
C C 510 IF (MP(IP1=1) = MP(IP1=2))520,520,530
C C YFS, REVERSING DIRECTION
C C 520 ICT = -1
C C GO TO 550
C C 530 NUMP = NUMP + IC
C C IF (ICT .GT. 0) GO TO 540
C C CHECK FOR INITIAL CUTTING PLANE
C C IF (NUMP .GE. MPIN) GO TO 550
C C HAVE COME BACK ACROSS INITIAL PLANE
C C CHANGE SIGN OF ICT AND INCREASE MPMAX
C C ICT = +1
C C NUMP = MPMAX + 1
C C MPIN = NUMP
C C 540 IF (NUMP .GT. MPMAX) MPMAX = NUMP
C C
C C CALCULATE INTERSECTION
C C 550 NPI(NUMP) = NPI(NUMP) + 1

```

MERID

MERID

MERI 351
 MERI 352
 MERI 353
 MERI 354
 MERI 355
 MERI 356
 MERI 357
 MERI 358
 MERI 359
 MERI 360
 MERI 361
 MERI 362
 MERI 363
 MERI 364
 MERI 365
 MERI 366
 MERI 367
 MERI 368
 MERI 369
 MERI 370
 MERI 371
 MERI 372
 MERI 373
 MERI 374
 MERI 375
 MERI 376
 MERI 377
 MERI 378
 MERI 379
 MERI 380
 MERI 381
 MERI 382
 MERI 383
 MERI 384
 MERI 385

```

NP = NPI(NUMP)
MPI(NUMP) = NMP
DY = YP(IP2) - YP(IP1)
DZ = ZP(IP2) - ZP(IP1)
CSP = COS(PI-PHI(NMP))
SNP = SIN(PI-PHI(NMP))

LR = 0.0
RI(NUMP, NP) = 0.0
DYDZ = DY**2 + DZ**2
IF (DYDZ .LE. 1.0E-6) GO TO 560
DYP = YP(IP1) - YPA(NMP)
DZP = ZP(IP1)
RIMAG = (DZP*DY - DYP*DZ)/(CSP*DY - SNP*DZ)
IF (INPHI .LT. 2) RIMAG = ARS(RIMAG)
RI(NUMP, NP) = RIMAG
YPI = SNP*RI(NUMP, NP) + YPA(NMP)
ZPI = CSP*RI(NUMP, NP)
LR = SORT(((YPI-YP(IP1))**2 + (ZPI-ZP(IP1))**2)/DYDZ)
PI(NUMP, NP) = ARS(RIMAG)

560 CONTINUE
XA(NUMP, NP) = XP(IP1) + (XP(IP2) - XP(IP1))*LR
XI(NUMP, NP) = XN(IP1) + (XN(IP2) - XN(IP1))*LR
YI(NUMP, NP) = YN(IP1) + (YN(IP2) - YN(IP1))*LR
ZI(NUMP, NP) = ZN(IP1) + (ZN(IP2) - ZN(IP1))*LR
NINT(NUMP) = NINT(NMP) + 1
PJ = JE
WRITE (TAPER) RJ, XI(NUMP, NP), YI(NUMP, NP), ZI(NUMP, NP), RI(NUMP, NP)
1 , XA(NUMP, NP)
MPTOT = MPOT + 1
NREC(NMP, NINT(NMP)) = MPTOT

```

C

C

C

C

C CONTINUE TO NEXT INTERSECTION

MERID

MERID

```
570 NMP = NMP + IMP
    IF (IMP)580,580,590
580 IF (MP(IP2) = NMP)530,440,540
590 IF (MP(IP2) = NMP)440,530,530
C
C
C ALL INTERSECTIONS FOUND ON CURRENT ELEMENT. SAVE THE RESULTS.
600 CONTINUE
    IF (MPMAX = 0) GO TO 650
    IF (IPRINT .EQ. 0) GO TO 650
    WRITE (TAPE07,610)
610 FORMAT(1H0, 4H ID, 714, 3HND., 724, 2HMP, 745, 1HX, 765, 1HY,
    1 785, 1HZ, 7105, 1HR, 7125, 1HA//)
C
    NO 620 K = 1, MPMAX
    NPIK = NPI(K)
620 WRITE (TAPE07,630)COMPID(JJ),JE,MPI(K);
    1 XI(K,IP), YI(K,IP), ZI(K,IP), PI(K,IP),
    2 XA(K,IP), IP = 1,NPIK)
630 FORMAT(1H , 1XA4, 6XI3, 7XI3, 4X, 5F20.6/1H , 28X5F20.6)
C
C
C RIGHT OUT ELEMENT CORNER POINTS
    WRITE (TAPE07,640) (I, XN(I), YN(I), ZN(I), J=1,4)
640 FORMAT(1H , 21XI3, 4X, 3F20.6)
C
C
C CHECK FOR SYMMETRY
650 IF (SYMFCT .NE. 0) GO TO 670
    IF (JSYM .EQ. 0) GO TO 670
    IF (JE .GT. 1L) GO TO 670
    NO 660 N = NP,4
    660 YN(N) = -YN(N)
```

MERI 386
MERI 387
MERI 388
MERI 389
MERI 390
MERI 391
MERI 392
MERI 393
MERI 394
MERI 395
MERI 396
MERI 397
MERI 398
MERI 399
MERI 400
MERI 401
MERI 402
MERI 403
MERI 404
MERI 405
MERI 406
MERI 407
MERI 408
MERI 409
MERI 410
MERI 411
MERI 412
MERI 413
MERI 414
MERI 415
MERI 416
MERI 417
MERI 418
MERI 419
MERI 420

MERID

MERID

MERI 421
MERI 422
MERI 423
MERI 424
MERI 425
MERI 426
MERI 427
MERI 428
MERI 429
MERI 430
MERI 431
MERI 432
MERI 433
MERI 434
MERI 435
MERI 436
MERI 437
MERI 438
MERI 439
MERI 440
MERI 441
MERI 442
MERI 443
MERI 444
MERI 445
MERI 446
MERI 447
MERI 448
MERI 449
MERI 450
MERI 451
MERI 452
MERI 453
MERI 454
MERI 455

MERID

```

C      JE = LL + J
C      GO TO 260
C
C      670 CONTINUE
C      675 CONTINUE
C
C      END OF PANEL DO LOOP
C
C      ARRANGE INTERSECTIONS AS PER MERIDIAN
C      FIRST CHECK IF ANY INTERSECTIONS FOUND
C      680 IF (MPTOT .GT. 0) GO TO 700
C      WRITE (TAPEOT,690)
C      690 FORMAT(1H0,41HND INTERSECTIONS FOUND FOR CONFIGURATION ,
C      1      /1H0, 16HCASF TERMINATED.)
C      STOP
C
C      700 CONTINUE
C      NF = NINT(1)
C      IF (INPHI .EQ. 0) NINT(1) = NINT(NPL1)
C      WRITE (TAPEC) COMPID, JJ, NT, NPL, MPTOT,
C      1      PSID, THFO, PHIO, XPO, YPO, ZPO, INPHI,
C      2      (PHI(I), I=1,NPL), (NINT(I), I=1,NPL), (YPA(I), I=1,NPL)
C      NINT(1) = NF
C      DO 810 I = 1, NPL
C      IF I = 1
C      710 NF = NINT(II)
C      IF (NF .FO. 0) GO TO 820
C      K = 0
C      K1 = 0
C      K2 = 1
C      K3 = 0
C      REWIND TAPEB
C
```

MERID

MERI 456
MERI 457
MERI 458
MERI 459
MERI 460
MERI 461
MERI 462
MERI 463
MERI 464
MERI 465
MERI 466
MERI 467
MERI 468
MERI 469
MERI 470
MERI 471
MERI 472
MERI 473
MERI 474
MERI 475
MERI 476
MERI 477
MERI 478
MERI 479
MERI 480
MERI 481
MERI 482
MERI 483
MERI 484
MERI 485
MERI 486
MERI 487
MERI 488
MERI 489
MERI 490

```
C ZERO OUT COMPONENT COUNTERS
  DD 720 L = 1,JJ
  720 NI(L) = 0
C
  DD 810 J = 1,NF
  NN = NREC(II,J)
  730 K = K + 1
  C READ (TAPEB,END=830) XREC
  READ (TAPEB) XREC
  IF (EOF(TAPER)) 830,731
  731 IF (K,NE. NN) GO TO 730
  IF (K3.EQ. 1) GO TO 750
C CHECK ON COMPONENT
  740 K1 = K1 + 1
  IF (K1.GT. JJ) GO TO 810
  NI(K1) = 0
  NT1 = NT(K1)
  IF (NT1.LE. 0) GO TO 740
C
  750 K2 = K2 + 1
  IF (K2.EQ. K1) GO TO 750
  IF (K2.LE. JJ) GO TO 760
  NT2 = MPTOT + 1
  GO TO 770
  760 NT2 = NT(K2)
  IF (NT2.EQ. 0) GO TO 750
  770 K3 = 0
C
  C TEST IF RECORD WITHIN RANGE
  780 IF ((NN.GE.NT1).AND.(NN.LT.NT2)) GO TO 790
C
  C NOT IN RANGE = GO TO NEXT COMPONENT
  GO TO 740
C
```

MERID

MERID

MERI 491
MERI 492
MERI 493
MERI 494
MERI 495
MERI 496
MERI 497
MERI 498
MERI 499
MERI 500
MERI 501
MERI 502
MERI 503
MERI 504
MERI 505
MERI 506
MERI 507
MERI 508
MERI 509
MERI 510
MERI 511
MERI 512
MERI 513
MERI 514
MERI 515

```
790 IF (K3 .EQ. 1) GO TO 800
    NI(K1) = J
    K3 = 1
C
800 WRITE (TAPEC) K1, XRFC
810 CONTINUE
C
820 ALL ELEMENTS IN CURRENT MERIDIAN STORED.
830 SAVE COMPONENTS COUNTERS
    WRITE (TAPEC) (NI(L),L=1,JJ)
C
840 IF (II .NE. 1) GO TO 830
    IF (INPHT .GT. 1) GO TO 830
    IF (NF .NE. 0) GO TO 830
    II = NPL1
    GO TO 710
850 CONTINUE
C
    RETURN
C
900 WRITE (TAPEC,910)
910 FORMAT (1H ,43H**PANEL NUMBER IS GREATER THAN THE MAXIMUM ,
: 34H STORED ON UNIT 10 ** PROGRAM STOP )
    STOP
    END
```

MERID

NCONE

```

C
C SURROUTINE NCONE(DCRI,EM,G, DSR, EMC, PCP1)
C THIS IS AN IMPROVED APPROXIMATE CONE FLOW SOLUTION
C DEVELOPED BY D. N. SMYTH
C SECOND ORDER THEORY USED FOR SMALL VALUES OF SIMILARITY PARAMETER
C APPROXIMATE SOLUTION OF HAMMITT AND MURTHY FOR LARGE VALUES
C MIDRANGE GIVEN BY SUITABLE TRANSITION FUNCTION
C
DCR = DCRI
ISDET = 0
RC = .17453292E+1
SINDC = SIN(DCR)
FMSIN = FM*SINDC
FMSQ = EM**2
KSOL = 2
C
C CALCULATE UPPER TRANSITION POINT, EMSINF.
FMSINF = 1.40
IF (FM.GE.10.0) GO TO 8
FMSINF = 0.375
IF (EM.GT.1.5) EMSINF = 1.40 - 1.075*EXP((EM-1.5)*(-0.8))
8 IF (EMSIN,LT,EMSINF) GO TO 20
KSOL = 1
C
C CALCULATE CONE SURFACE PROPERTIES USING METHOD OF
C HAMMITT AND MURTHY.
9 SINSQ = SINDC**2
SIN2 = SIN(2.*DCR)
M2 = (G-1.)*SINSQ + 2./EMSQ
M1 = 2. * (G+5.)*SINSQ
MX = M2*M1/SIN2**2
IF (ABS(M1).LE.1.E-3) GO TO 10
RAD = SQRT(1.0 + MX)
DSR = DCR - SIN2*(1.0 - RAD)/M1

```

NCON 001
 NCON 002
 NCON 003
 NCON 004
 NCON 005
 NCON 006
 NCON 007
 NCON 008
 NCON 009
 NCON 010
 NCON 011
 NCON 012
 NCON 013
 NCON 014
 NCON 015
 NCON 016
 NCON 017
 NCON 018
 NCON 019
 NCON 020
 NCON 021
 NCON 022
 NCON 023
 NCON 024
 NCON 025
 NCON 026
 NCON 027
 NCON 028
 NCON 029
 NCON 030
 NCON 031
 NCON 032
 NCON 033
 NCON 034
 NCON 035

NCONE

NCONE

```

GO TO 11
10 DSP = DCP + 0.5*H2/SIN2*(1.0 - 0.25*HX*(1.0 - 0.5*HX))
11 EMSQ = EMSQ*SIN(DSR)**2
HS2 = (DSR - DCR)**2
FWC = SQRT((FMSQ*EMSSQ)*(1.0 + 2.0*HS2)/(1.0 + 0.5*(G - 1.0)*
1 (EMSSQ*(1.0 + 2.0*HS2) - 2.0*HS2*FMSQ))
FLIM = (G - 1.0)/(G + 1.0)
PSP1 = 2.0*G/(G+1.0)*EMSSQ - FLIM
TYS = (PSP1 + FLIM)/(1.0 + FLIM*PSF1)/PSP1
TCT1 = (1.0 + (G-1.0)*0.5*EMSQ)/(1.0 + (G-1.0)*0.5*EMC**2)
PCPS = (TCT1+TITS)**(G/(G-1.))
PCPI = PCPS*PSP1
CP = 2.0*(PCPI - 1.0)/(G*EMSQ)
GO TO (40,31), KSQL

C
C
C CALCULATE LOWER TRANSITION POINT, EMSINO.
20 EMSINO = 0.2
IF (FMSQ.GT.3.0) EMSINO = 0.3
IF (FMSIN.GT.EMSINO) GO TO 30
KSQL = 1

C
C
C CALCULATE CONE SURFACE PROPERTIES USING 2ND-ORDER THEORY.
21 BETA = SQRT(FMSQ - 1.)
TANOC = TAN(DCR)
TANSQ = TANOC**2
BETA2 = 5.0*FMSQ - 1.0
AM = 3.25*EMSQ + 0.5 + (G+1.0)*(EMSQ/BETA)**2
PHI = 0.69314718 - ALOG(BETA) - ALOG(TANOC)
CP = TANSQ*(2.*PHI-1.0)+TANSQ*(3.+(BETA*PHI)**2-BETA*2*PHI + AM)
IF (KSQL.EQ.2) GO TO 22
IF (TSD.T.FG.2) RETURN
DSR = ARSIN(1./EM)
PCPI = G*EMSQ/CP *0.5 + 1.0
TCT1 = PCPI**((G-1.)/G)

```

NCON 036
 NCON 037
 NCON 038
 NCON 039
 NCON 040
 NCON 041
 NCON 042
 NCON 043
 NCON 044
 NCON 045
 NCON 046
 NCON 047
 NCON 048
 NCON 049
 NCON 050
 NCON 051
 NCON 052
 NCON 053
 NCON 054
 NCON 055
 NCON 056
 NCON 057
 NCON 058
 NCON 059
 NCON 060
 NCON 061
 NCON 062
 NCON 063
 NCON 064
 NCON 065
 NCON 066
 NCON 067
 NCON 068
 NCON 069
 NCON 070

NCONE

NCONE

```

EMC = SQRT(2./(G-1.))*((1.0+(G-1.)*0.5*EMSQ)/TCT1 - 1.0)
GO TO 40
22 DCP2 = IANDC*(4.*(PHI - 1.0) + IANSQ*(4.*PHI*(2.*PHI-1.0)*
1 BETA**2 - BETA PHI - 1.0) + 4.*A**2)/(EM*CCS(CCR)**3)
GO TO 32
C
C CALCULATE CONF SURFACE USING TRANSITION SOLUTION
C (THAT IS - A COMBINATION OF FIRST- AND SECOND-ORDER AND HAMMITT-MURPHY).
30 DXF = EMSINF - EMSINO
DX = FMSIN - EMSINO
SINDC = EMSINF/FM
DCP = ARSIN(SINDC)
GO TO 9
31 CPHM = CP
DCP = ARSIN(FMSINO/EM)
GO TO 21
32 A = (CPHM - CP - DCP2*DXF)/DXF**2
CP = CP + DX*(DCP2 + 4*DX)
IF (ISDET.EQ.2) RETURN
PCP1 = 0.5*G*EMSQ*CP + 1.0
FMSNF = FMSIN(DSR)
FMS = (FMSNF - 1.)*(FMSIN/EMSINF)**2 + 1.0
EMNSSQ = EMNS**2
EMSSQ = ((G+1.)*FM*EMNS)**2 = 4.*(EMNSSQ-1.)*(G*EMNSSQ+1.)
DSQ = ARSIN(FMNS/EM)
FMSSQ = EMSSQ/((2.*G*EMNSSQ*(G-1.))+(G-1.)*FMNSSQ+2.)
PSPC = (2.*G*EMNSSQ - (G-1.))/(G+1.)/PCP1
EMC = SQRT(((1.+0.5*(G-1.)*EMSQ)*PSPC**2+(G-1.)/G) - 1.)*2/(G-1.)
TCT1 = (1.+(G-1.)*0.5*FMNSQ)/(1.+(G-1.)*0.5*EMC**2)
40 CONTINUE
RETURN
END

```

NCON 071
NCON 072
NCON 073
NCON 074
NCON 075
NCON 076
NCON 077
NCON 078
NCON 079
NCON 080
NCON 081
NCON 082
NCON 083
NCON 084
NCON 085
NCON 086
NCON 087
NCON 088
NCON 089
NCON 090
NCON 091
NCON 092
NCON 093
NCON 094
NCON 095
NCON 096
NCON 097
NCON 098
NCON 099
NCON 100
NCON 101
NCON 102

NCONE

CONEA

```

SURROUTINE CONEA(CDR,EM,G,ALF,PHI,ALFP,EMP,DSR,FMSURF,PSURF,IT)
C CIRCULAR CONE AT ANGLE OF ATTACK.
C BASED ON ARC CP #792 AND EXTENDED BY D.N. SMYTH.
C
C DIMENSION FSR(8), BS(8), ANGLE(3)
C DIMENSION AAR(6), A(18)
C
C COMMON/FCBS/ FS,RS
C
C DATA A/ -0.07657, 1.4775, 0.184669,
2 0.42339, 0.13241,0.034871,
3 -0.002083,-.075795,-.01923,
4 0.29891, -.10011,0.29589,
5 -0.99727, -.41751,0.068791,
6 -.039442, 0.10422,0.063801/
C
GM1 = G - 1.0
CG1 = G + 1.0
PMTN = 0.3
CPCONE = 0.0
CPSTAG = 0.0
CPX = 0.0
MPSK = 0.0
COSA = COS(ALF)
SINA = SIN(ALF)
CDSP = COS(PHI)
SINP = SIN(PHI)
FM2 = FM1*2
CPMIN = (PMIN - 1.0)/(0.5*G*EM2)
C
IF (CDR .LT. 0.0) GO TO 35
IF (CDR .LT. 1.E-4) GO TO 5
IF (CDR .LT. 0.0872665) GO TO 4
IF (TT .EQ. 1) GO TO 4
C

```

```

CONE 001
CONE 002
CONE 003
CONE 004
CONE 005
CONE 006
CONE 007
CONE 008
CONE 009
CONE 010
CONE 011
CONE 012
CONE 013
CONE 014
CONE 015
CONE 016
CONE 017
CONE 018
CONE 019
CONE 020
CONE 021
CONE 022
CONE 023
CONE 024
CONE 025
CONE 026
CONE 027
CONE 028
CONE 029
CONE 030
CONE 031
CONE 032
CONE 033
CONE 034
CONE 035

```

CONEA

CONEA

```

C JONES METHOD (AIAA JL., FEB, 1972, PP234-236).
  FMPI = 1.0/EM2
  COS2P = COS(P.*PHI)
  T = SIN(2.0*DCR)*TAN(DCR)
  AC = ALF/DCR
  FMA = FM
  CALL NCDNE(DCR,EM,G,DSC,EMS,PCONE)
  CPONE = (PCONE = 1.0)/(0.5*G*EM2)

C      DO 3 I = 1,6
      J = 3*(I-1) + 1
      AA(I) = A(J) + A(J+1)*COSP + A(J+2)*COS2P

C      CPX = (AA(1)*T + AA(2)*T*EM2I + AA(3)*FMPI)*AC
      1  + (AA(4)*T + AA(5)*T*EM2I + AA(6)*FMPI)*AC*P
      GO TO 40

C
C      AXIAL FLOW COMPONENT
      4 FMA = EM*COXA
      CALL NCDNE(DCR,EMA,G,DSC,EMS,PCONE)
      CPONE = (PCONE = 1.0)/(0.5*G*EM2)
      5 IF (ALF.LT. 1.E-4) GO TO 40

C
C      NORMAL FLOW COMPONENT
      EMC2 = EM2*(SINA)**2
      IF (EMC2 = 1.0) 10, 10, 20
C      SUBSONIC NORMAL COMPONENT
      10 PSTAG = (1.0 + 0.5*GM1*EMC2)**(G/GM1)
      GO TO 30
C      SUPERSONIC NORMAL COMPONENT
      20 PSTAG = (0.5*GM1*EMC2)**(G/GM1)*(GM1/(2.*G*EMC2-GM1))**((1./GM1)
C      30 CPSTAG = (PSTAG + 1.0)/(0.5*G*EM2)*COSB*ABS(COSB)

```

```

CONF 036
CONF 037
CONF 038
CONF 039
CONF 040
CONF 041
CONF 042
CONF 043
CONF 044
CONF 045
CONF 046
CONF 047
CONF 048
CONF 049
CONF 050
CONF 051
CONF 052
CONF 053
CONF 054
CONF 055
CONF 056
CONF 057
CONF 058
CONF 059
CONF 060
CONF 061
CONF 062
CONF 063
CONF 064
CONF 065
CONF 066
CONF 067
CONF 068
CONF 069
CONF 070

```

CONEA

CONEA

CONE 071
CONE 072
CONE 073
CONE 074
CONE 075
CONE 076
CONE 077
CONE 078
CONE 079
CONE 080
CONE 081
CONE 082
CONE 083
CONE 084
CONE 085
CONE 086
CONE 087
CONE 088
CONE 089
CONE 090
CONE 091
CONE 092
CONE 093
CONE 094
CONE 095
CONE 096
CONE 097
CONE 098
CONE 099
CONE 100
CONE 101
CONE 102
CONE 103
CONE 104
CONE 105

```
IF (CPSTAG .LT. CPMIN) CPSTAG = CPMIN
IF (OCR .LT. 1.E-4) GO TO 40
C CROSS PRODUCT TERM = EMPIRICAL FIT FROM CP #792.
SCD = SIN(OCR)*COS(DCR)
X = 0.5*3.1415927/(SCD*SQRT(EM2-1.0))
RK = 1.95 + 0.07*COS(X)
CPX = 2.0*RK*SCD*COSP*SINA*COXA
GO TO 40

C
C CONVE FLOW IS DETACHED OR UNDEFINED (OCR .LT. 0.0)
C USE IMPACT METHODS
C
C FIRST GFT IMPACT ANGLE (IN MERIDIAN PLANE)
35 CONTINUE
DELI = ALFP + OCR
IF (DELI .GT. 0.0) GO TO 36

C
C EXPANSION FLOW
ANGLF(2) = ARS(DFLI)*57.29577951
FS(6) = EMP
IPRINT = 0
ISDET = 2
CALL EXPAND(ANGLE, MER, IPRINT, ISDET, CP)
EMJSURF = RS(6)
FM2 = FMP**2
PSURF = 0.5*G*EM2*CP + 1.0
IF (PSURF .LT. PMIN) PSURF = PMIN
GO TO 50

C
C COMPRESSION FLOW, USE TANGENT-CONE EMPIRICAL
36 CK = 2.*GPI/(G+3.)*EMP*SIN(DFLI)
EMNS = (CK + EXP(-CK))**2
CP = (8.*GPI*EMNS/((3.+5.)*EMNS + 2.))*SIN(DFLI)**2
PSURF = 0.5*G*EM2*CP + 1.0
IF (PSURF .LT. PMIN) PSURF = PMIN
```

CONEA

CONEA

```

C      PCONE = PSURF
C      DPSK = PCONE * (2.*G*EMNS + GM1)/GPI
C      GO TO 50
C
C      COMBINED SURFACE PRESSURE
C      40 PSURF = 0.5*G*FM2*(CPCONE + CPSTAG + CPX) + 1.0
C
C      LIMIT MINIMUM PSURF
C      IF (PSURF .LT. PMIN) PSURF = PMIN
C
C      SHOCK ANGLE DSR, RELATIVE TO AXIS
C
C      USE IMPACT METHODS TO OBTAIN DPSK
C      (ADJUST TO ALPHA = ZERO RESULTS)
C
C      DELI = DCR
C      IS = 1
C      FDP = 1.0
C      EMC = EMA
C      GO TO 46
C
C      CALCULATE VALUE FROM NCONE
C      45 CONTINUE
C      EMNS = (EMA+SIN(DSC))**2
C      PS = (2.*G*EMNS + GM1)/GPI
C      FDP = PS/P2
C      EMC = EMP
C
C      NOW CALCULATE VALUE AT ACTUAL IMPACT ANGLE
C      DELI = ALFP + DCR
C      DPSK = 0.0
C      IF (DELI .LE. 0.0) GO TO 50
C      IS = 2
C      46 EMD = EMC+SIN(DELI)
C      CK = 2.*GPI/(G+3.)*EMD

```

106 CONE
107 CONE
108 CONE
109 CONE
110 CONE
111 CONE
112 CONE
113 CONE
114 CONE
115 CONE
116 CONE
117 CONE
118 CONE
119 CONE
120 CONE
121 CONE
122 CONE
123 CONE
124 CONE
125 CONE
126 CONE
127 CONE
128 CONE
129 CONE
130 CONE
131 CONE
132 CONE
133 CONE
134 CONE
135 CONE
136 CONE
137 CONE
138 CONE
139 CONE
140 CONE

CONEA

CONEA

CONE 141
CONE 142
CONE 143
CONE 144
CONE 145
CONE 146
CONE 147
CONE 148
CONE 149
CONE 150
CONE 151
CONE 152
CONE 153
CONE 154
CONE 155
CONE 156
CONE 157
CONE 158
CONE 159
CONE 160
CONE 161

```
FMNS = (CK + EXP(=CK))**2
P2 = (2.*G*EMNS + GM1)/GPI
GO TO (45,47), IS
47 DPSK = PSURF - P2*FDP
50 IF (DPSK .LT. 0.0) DPSK = 0.0
PSHK = PSURF - DPSK
IF (PSHK .LT. 1.0) PSHK = 1.0
C SURFACE MACH NUMBER
EMSURF = SQRT((EM2*(GPI*PSHK+GM1) - 2.*GPI*GM1)
1 / (PSHK*(GM1*PSHK + GPI)))
C
FMP = FMP**2
DSR = (GPI*PSHK + GM1)/(2.*G*EM2)
IF (DSR .GT. 1.0) DSR = 1.0
IF (DSR .LT. 0.0) DSR = 1.0/EM2
DSR = ARSIN(SQRT(DSR))
DSR = DSR - ALFP
C
C RETURN
END
```

CONEA

MOUT

```

SURROUTINE MOUT
COMMON /FXFC/ CASE,ITITLE(15), PAGE, ERROR
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEFC,TAPEFD,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
DIMENSION PHI(40), NINT(40), COMPID(10), NI(10), NE(10),
2 RJ(50), X(50), Y(50), Z(50), R(50), A(50), YPA(40), AX(50),
3 J(50), JX(50), IOR(10), ID(50), DORD(4)
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
INTEGER PAGE, ERROR
DATA RC/1.74532925E-2/
DATA DORD/1HA, 1HX, 1HY, 1HZ/
IOV = 0
IOR = -1
COMPR = 1
IPRINT = 1
DO 9 I=1,10
9 IOR(I) = 0
DO 20 I = 1,40
PHI(I) = 0.0
YPA(I) = 0.0
20 CONTINUE
REWIND TAPEC
REWIND TAPEB
C IOR(I) = FLAG TO ORDER INDIVIDUAL COMPONENT (RANDOM)
C IOG = FLAG TO ORDER COMPONENTS BY GROUPS
C COMPR = BASE COMPONENT (OTHERS ADDED TO THIS ONE)
C
C POINTS FOR A PARTICULAR MERIDIAN ARE BROUGHT INTO CORE
C INDIVIDUAL COMPONENTS ARE THEN ORDERED (IF REQUIRED, NON-ZERO IOR)
C NEXT, GROUPS ARE ORDERED (IF REQUIRED, NON-ZERO, POSITIVE IOG)
C NOTE, A NEGATIVE IOG WILL CAUSE RANDOM ORDERING OF THE WHOLE SET.

```

```

MOUT 001
MOUT 002
MOUT 003
MOUT 004
MOUT 005
MOUT 006
MOUT 007
MOUT 008
MOUT 009
MOUT 010
MOUT 011
MOUT 012
MOUT 013
MOUT 014
MOUT 015
MOUT 016
MOUT 017
MOUT 018
MOUT 019
MOUT 020
MOUT 021
MOUT 022
MOUT 023
MOUT 024
MOUT 025
MOUT 026
MOUT 027
MOUT 028
MOUT 029
MOUT 030
MOUT 031
MOUT 032
MOUT 033
MOUT 034
MOUT 035

```

MOUT

MOUT

```

C      READ(TAPE6)      COMPID, NC, NE, NPL, MPINT,
1      PSIO, THFO, PHIO, XPO, YPO, ZPO, ITP1,
2      (PHI(I), I=1,NPL), (NINT(I), I=1,NPL), (YPA(I), I=1,NPL)
C
C      WRITE(TAPE6) COMPID, NC, NE, NPL, MPINT,
1      PSIO, THFO, PHIO, XPO, YPO, ZPO, ITP1,
2      (PHI(I), I=1,NPL), (YPA(I), I=1,NPL)
C
C      IF (IPRINT .EQ. 1)
1WRITE (TAPE7,30) NPL,MPINT,
2 PHIO, PSIO, THFO, XPO, YPO, ZPO
30 FORMAT(1H1, T10,15HCONFIGURATION , T40,
1 174NUMBER OF CUTS = , I2, T40,15HTOTAL POINTS = , I4/1H0,
2 7HPHIO = , F11.6, 5X, T40, 7HPSIO = , F11.6, 5X,
3 7HTHED = , F11.6, 5X, T40, 5HXO = , F11.6, 5X,
4 5HYO = , F11.6, 5X, T40, 5HZO = , F11.6, 5X)
C
C      DO 500 KKE 1,NPL
C      RPING MERIDIAN DATA INTO CORE
NF = NINT(KK)
NT = NF
NTA = NT
NFL = 0
PHIO = PHI(KK)/RC
IF (IPRINT .EQ. 1)
1WRITE (TAPE7,40) KK,PHIO, YPA(KK), NINT(KK)
40 FORMAT(1H0,13HPLANE NUMBER , I2, T20,6HPHI = , F11.6, 6X,
1 5HYPA = , F11.6,
2 T71, 26HNUMBER OF INTERSECTIONS = , I3/)
IF (NF .EQ. 0) GO TO 500
DO 50 1 = 1,NF

```

MOUT

MOUT

```

      J(I) = I
      JX(I) = I
50 REAC (TAPEC) I(I), RJ(I), X(I), Y(I), Z(I), R(I), AX(I)
C
C READ IN COMPONENT COUNTERS
  READ (TAPEC) (NI(I),I=1,NC)
C
C
C   IF (IOV .GT. 0) GO TO (70,90,110), IOV
C
C ORDERING WILL USE AXIAL COORD.
  DO 60 I = 1,NF
60 A(I) = AX(I)
  ORD = DORD(1)
  GO TO 130
C
C ORDERING WILL USE X COORD.
  DO 70 I = 1,NF
70 A(I) = X(I)
  ORD = DORD(2)
  GO TO 130
C
C ORDERING WILL USE Y COORD.
  DO 90 I = 1,NF
90 A(I) = Y(I)
  ORD = DORD(3)
  GO TO 130
C
C ORDERING WILL USE Z COORD.
  DO 110 I = 1,NF
110 A(I) = Z(I)
  ORD = DORD(4)
C
C 130 CONTINUE

```

MOUT 071
MOUT 072
MOUT 073
MOUT 074
MOUT 075
MOUT 076
MOUT 077
MOUT 078
MOUT 079
MOUT 080
MOUT 081
MOUT 082
MOUT 083
MOUT 084
MOUT 085
MOUT 086
MOUT 087
MOUT 088
MOUT 089
MOUT 090
MOUT 091
MOUT 092
MOUT 093
MOUT 094
MOUT 095
MOUT 096
MOUT 097
MOUT 098
MOUT 099
MOUT 100
MOUT 101
MOUT 102
MOUT 103
MOUT 104
MOUT 105

MOUT

MOUT

MOUT 106
MOUT 107
MOUT 108
MOUT 109
MOUT 110
MOUT 111
MOUT 112
MOUT 113
MOUT 114
MOUT 115
MOUT 116
MOUT 117
MOUT 118
MOUT 119
MOUT 120
MOUT 121
MOUT 122
MOUT 123
MOUT 124
MOUT 125
MOUT 126
MOUT 127
MOUT 128
MOUT 129
MOUT 130
MOUT 131
MOUT 132
MOUT 133
MOUT 134
MOUT 135
MOUT 136
MOUT 137
MOUT 138
MOUT 139
MOUT 140

C FIRST ORDER RANDOM SETS (W,R,T, AXIAL CONTR. A)

```

N1 = 1
I = NC
IF (IOS .LT. 0) GO TO 160
I = 0
140 I = I + 1
IF (TOR(I) .EQ. 0) GO TO 230
N1 = NI(I)
IF (N1 .EQ. 0) GO TO 230
IF (I - NC)150,160,160
150 I2 = I
155 I2 = I2 + 1
IF (I2 .GT. NC) GO TO 160
NT = NI(I2) = 1
IF (NT) 155, 155, 170
160 NT = NF
170 NT1 = NT - 1
DO 220 L = N1,NT1
J1 = J(L)
IF (J1 .EQ. 0) GO TO 230
L1 = L + 1
C
C 210 K = L1,NTA
JL = J(K)
IF (J2 .EQ. 0) GO TO 215
DELA = A(J1) = A(J2)
IF (DELA) 190,190,185
C
185 IF (ABS(DELA) .GT. (ABS(A(J1)) + 1.0)*0.001) GO TO 210
C
190 KD = K - L
K1 = K
DO 200 II = 1,KD
K2 = K1 - 1
J(K1) = J(K2)

```

MOUT

MOUT

```
200 K1 = K2
   J(I) = J2
C
   IF (ABS(DELA) .GT. (ABS(A(J:)) + 1.0)*0.001) GO TO 205
C TWO EQUAL POINTS. ELIMINATE J1 FROM SET.
   NT = NT - 1
   NEL = NEL + 1
   DO 204 II = L1,NT
     L2 = II + 1
204 J(II) = J(L2)
     J(NT + 1) = 0
     J1 = J2
     GO TO 180
205 J1 = J2
210 CONTINUE
215 NTA = NT
220 CONTINUE
230 NT = NTA
   IF (I .LT. NC) GO TO 140
C
C CHECK IF GROUP ORDERING REQUIRED
   IF (IOO .GT. 0) GO TO 236
   L = 0
   DO 235 I = 1,NT
     L = L + 1
235 JX(L) = J(I)
     GO TO 460
C
C NOW ORDER GROUPS
C FIND BASE COMPONENT
236 DO 240 I = 1,NC
   IB = I
   IF (COMPID(I) .EQ. COMPB) GO TO 250
```

MOUT 141
MOUT 142
MOUT 143
MOUT 144
MOUT 145
MOUT 146
MOUT 147
MOUT 148
MOUT 149
MOUT 150
MOUT 151
MOUT 152
MOUT 153
MOUT 154
MOUT 155
MOUT 156
MOUT 157
MOUT 158
MOUT 159
MOUT 160
MOUT 161
MOUT 162
MOUT 163
MOUT 164
MOUT 165
MOUT 166
MOUT 167
MOUT 168
MOUT 169
MOUT 170
MOUT 171
MOUT 172
MOUT 173
MOUT 174
MOUT 175

MOUT

MOUT

MOUT 176
MOUT 177
MOUT 178
MOUT 179
MOUT 180
MOUT 181
MOUT 182
MOUT 183
MOUT 184
MOUT 185
MOUT 186
MOUT 187
MOUT 188
MOUT 189
MOUT 190
MOUT 191
MOUT 192
MOUT 193
MOUT 194
MOUT 195
MOUT 196
MOUT 197
MOUT 198
MOUT 199
MOUT 200
MOUT 201
MOUT 202
MOUT 203
MOUT 204
MOUT 205
MOUT 206
MOUT 207
MOUT 208
MOUT 209
MOUT 210

MOUT

```

240 CONTINUE
  IB = 1
C IDENTIFY ELEMENTS OF BASE COMPONENT
250 NI = NI(18)
  IF (NI .GT. 0) GO TO 260
  NI = 1
  GO TO 290
260 CONTINUE
  IF (18 = NC)270,290,290
270 181 = 18
280 181 = 181 + 1
  IF (181 .GT. NC) GO TO 290
  N2 = NI(181) + 1
  IF (N2 .LE. 0) GO TO 280
  GO TO 300
290 N2 = NF
300 NT = N2 - N1 + 1
  L = 0
  DO 310 I = N1,N2
  L = L + 1
310 JX(L) = J(I)
  IF (NC .LE. 1) GO TO 460
  IF (NI(18) .EQ. 0) GO TO 460
C NOW CYCLE OTHER COMPONENTS
  DO 450 IC = 1,NC
  IF (IC .EQ. 18) GO TO 450
  NCI = NI(IC)
  IF (NC1 .EQ. 0) GO TO 450
  IF (IC = NC)320,340,340
320 IC1 = IC
330 IC1 = IC1 + 1
  IF (IC1 .GT. NC) GO TO 340
  NCP = NI(IC1) + 1

```

MOUT

MOUT 211
MOUT 212
MOUT 213
MOUT 214
MOUT 215
MOUT 216
MOUT 217
MOUT 218
MOUT 219
MOUT 220
MOUT 221
MOUT 222
MOUT 223
MOUT 224
MOUT 225
MOUT 226
MOUT 227
MOUT 228
MOUT 229
MOUT 230
MOUT 231
MOUT 232
MOUT 233
MOUT 234
MOUT 235
MOUT 236
MOUT 237
MOUT 238
MOUT 239
MOUT 240
MOUT 241
MOUT 242
MOUT 243
MOUT 244
MOUT 245

MOUT

```
IF (NC2 .LE. 0) GO TO 330
GO TO 350
340 NC2 = NF
C
350 DO 360 I = NC1,NC2
L = L + 1
360 JX(L) = J(I)
IF (IOG .LE. 0) GO TO 450
LT = L
J1 = J(NC1)
DO 370 LX = 1,NT
L1 = LX
L2 = L1 + 1
J2 = JX(LX)
IF (A(J1) = A(J2))370,390,380
370 CONTINUE
C
C ELEMENTS DO NOT OVERLAP
NT = LT
GO TO 450
C
C SAVE REMAINING INDICES FROM BASE COMPONENT
380 L2 = L1
IF (L2 .GT. NT) GO TO 410
390 LJ1 = JX(L2)
LJ2 = J(NC2)
IF (A(LJ1) .EQ. A(LJ2)) NC2 = NC2 - 1
IX = NI - 1
DO 400 LX = L2,NT
IX = IX + 1
400 J(TX) = JX(LX)
C
C INSERT ELEMENTS OF COMPONENT IC
410 DO 420 I = NC1,NC2
JX(L1) = J(I)
```

MOUT

```

C      420 LI = LI + 1
C      C  FILL IN REMAINING BASE ELEMENTS
      IF (L2 .GT. NT) GO TO 440
      DO 430 I = N1,IX
      JX(LI) = J(I)
      430 LI = LI + 1
C
C      RESFY POINT NUMBERS
      440 L = LI - 1
      NT = L
C
C      450 CONTINUE
      460 CONTINUE
C
C      IF (IPRINT .EQ. 1)
      1WRITE (TAPEOT,470) ORD,IOG, (IOR(I),I=1,10)
      470 FORMAT(100,10)POINTS ORDERED FOR , A1, 14H AS PER IOG = , I3,
      1 14H AND IOR(I) = , 10I3/140, T14, 2HIP, 7XIMJ, T42,14X,
      2 T62,14Y, T82,14Z, T102,14P, T122,14A/)
C
C      SAVE THE DATA ON TAPE
      WRITE(TAPEB) NT
C
C      DO 490 L = 1,NT
      I = JX(L)
      JJ = RJ(I)
      IC = ID(I)
      WRITE(TAPEB) IC,COMPID(IC),JJ,X(I),Y(I),Z(I),R(I),A(I)
      IF (IPRINT .EQ. 1)
      1WRITE (TAPEOT,480) COMPID(IC),JJ,X(I),Y(I),Z(I),R(I),AX(I)
      480 FORMAT(1H , T12,A4, 4X,5F20.6)
      490 CONTINUE
C
      IF (IPRINT .EQ. 1)

```

MOUT

MOUT

```
1WRITE(TAPEOT,495) NEL  
495 FORMAT(1H0, 13, 40H DUPLICATE POINTS REMOVED FROM THE DATA.)
```

C

C 500 CONTINUE

C

```
RETURN  
END
```

```
MOUT 281  
MOUT 282  
MOUT 283  
MOUT 284  
MOUT 285  
MOUT 286  
MOUT 287  
MOUT 288
```

MOUT

CADA

CADA 001
 CADA 002
 CADA 003
 CADA 004
 CADA 005
 CADA 006
 CADA 007
 CADA 008
 CADA 009
 CADA 010
 CADA 011
 CADA 012
 CADA 013
 CADA 014
 CADA 015
 CADA 016
 CADA 017
 CADA 018
 CADA 019
 CADA 020
 CADA 021
 CADA 022
 CADA 023
 CADA 024
 CADA 025
 CADA 026
 CADA 027
 CADA 028
 CADA 029
 CADA 030
 CADA 031
 CADA 032
 CADA 033
 CADA 034
 CADA 035

```

SURROUTINE CADA(M,X,Y,T,S,C)
  REVISED VERSION OF CIRCULAR ARC DATA ANALYSIS INC PLENTY OF OPTION
  AL CHECK OUT STUFF. IT CALCS FROM M(X,Y) PTS THE ARC LENGTH S, ANGL
  F T, AND CURVATURE C.
  DIMENSION X( 1),Y( 1),T( 1),S( 1),C( 1)
  FIRST SOME CONSTS,
  Z1=1.000
  Z2=2.000
  PI=3.14159265359
  ZR=180.000/PI
  ZT=0.000002
  M1=M+1
  NOW DO MAIN LOOP
  DO 60 J=2,M1
  X12=X(J)-X(J-1)
  Y12=Y(J)-Y(J-1)
  X23=X(J+1)-X(J)
  Y23=Y(J+1)-Y(J)
  X13=X(J+1)-X(J-1)
  Y13=Y(J+1)-Y(J-1)
  A12= ATAN2(Y12,X12)
  IF((X12.LT.0.000).AND.(Y12.LT.0.000)) A12=Z2*PI+A12
  A23= ATAN2(Y23,X23)
  IF((X23.LT.0.000).AND.(Y23.LT.0.000)) A23=Z2*PI+A23
  A13= ATAN2(Y13,X13)
  IF((X13.LT.0.000).AND.(Y13.LT.0.000)) A13=Z2*PI+A13
  DL12=Z2*(A23-A12)
  DL23=Z2*(A13-A12)
  DL13=Z2*(A23-A12)
  
```

CADA

CADA

CADA 036
 CADA 037
 CADA 038
 CADA 039
 CADA 040
 CADA 041
 CADA 042
 CADA 043
 CADA 044
 CADA 045
 CADA 046
 CADA 047
 CADA 048
 CADA 049
 CADA 050
 CADA 051
 CADA 052
 CADA 053
 CADA 054
 CADA 055
 CADA 056
 CADA 057
 CADA 058
 CADA 059
 CADA 060
 CADA 061
 CADA 062
 CADA 063
 CADA 064
 CADA 065
 CADA 066
 CADA 067
 CADA 068
 CADA 069
 CADA 070

```

T1=A12-A23+A13
T2=A23-A13+A12
T3=A13-A12+A23
D12= SORT(X12*X12+Y12*Y12)
D23= SORT(X23*X23+Y23*Y23)
F12=Z1
IF(DL12.LT.ZT) GO TO 10
V12=DL12/Z2
F12=V12/ SIN(V12)
10 S12=D12+F12
F23=Z1
IF(DL23.LT.ZT) GO TO 20
V23=DL23/Z2
F23=V23/ SIN(V23)
20 S23=D23+F23
C13=D13/(S12+S23)
IF(J.NE.2) GO TO 30
T(1)=T1
S(1)=0.000
C(1)=C13
30 T(J)=T2
S(J)=S(J-1)+S12
C(J)=C13
IF(J.NE.M1) GO TO 40
T(M)=T3
S(M)=S(M1)+S23
C(M)=C(M1)
40 T1D=T1+ZR
T2D=T2+ZR
T3D=T3+ZR
AD12=A12+ZR
AD23=A23+ZR
AD13=A13+ZR
60 CONTINUE
RETURN
  
```

CADA

CADA

CADA 071

CADA

END

```

C SURROUTINE FFSPEC
C THIS ROUTINE IS USED TO PERMIT THE GENERATION AND STORAGE OF
C UNIFORM FLOW FIELDS USING THE SIMPLE COMPRESSION, EXPANSION, ETC.,
C FEATURES OF THE PROGRAM.
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/EACH,ALT,PSYAG,VTSTAG,V,RENO,PFS,AFS,RHDFS,VIS
COMMON /TAPE/TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPEJ,TAPEK,TAPEL,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /FSBS/FS,BS
COMMON /FFIELD/TITLEM(10),TTITLES(10),TTITLEA(10),TTITLER(10),
1 IMTAR(9),IDTYP(5),LOSET(5),LCAR(20),LORG(20),L,
2 DIMF(6),IRW,LCNEXT,IREG,MACH,NAB,ALPHA,BETA,NREG,
3 NSREG,ITFLAG
DIMENSION TTITLE(15),FS(8),BS(8),ANGLE(3)
INTEGER ERROR,CASE
INTEGER TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPEJ,TAPEK,TAPEL,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
REAL MACH

C
C READ (TAPEIN,10) ISORCE
C FORMAT (I1)
C ITAB = ISORCF
C IF (ITAB.LT.1) .OR. ITAB.GT.4) GO TO 900
C
C READ (TAPEIN,20) DINF
C FORMAT (6F10.0)
C
C INITIALIZE FLAGS
C MEP = 0
C IPRINT = 0
C CPSTAG = 2.0

```

FFSP 036
 FFSP 037
 FFSP 038
 FFSP 039
 FFSP 040
 FFSP 041
 FFSP 042
 FFSP 043
 FFSP 044
 FFSP 045
 FFSP 046
 FFSP 047
 FFSP 048
 FFSP 049
 FFSP 050
 FFSP 051
 FFSP 052
 FFSP 053
 FFSP 054
 FFSP 055
 FFSP 056
 FFSP 057
 FFSP 058
 FFSP 059
 FFSP 060
 FFSP 061
 FFSP 062
 FFSP 063
 FFSP 064
 FFSP 065
 FFSP 066
 FFSP 067
 FFSP 068
 FFSP 069
 FFSP 070

```

ISDET = 0
IFIRST = 0
ICTYP(2) = 1
ANGLF(2) = ABS(DINF(5))

C INITIALIZE FREE STREAM DATA ARRAY
C
FS(1) = RHOF5
FS(2) = PFS
FS(3) = TFS
FS(4) = AFS
FS(5) = VIS
FS(6) = DINF(1)
FS(7) = DINF(1) * FS(4)
FS(8) = FS(1) * FS(7)/FS(5)

C
C
GO TO (210,220,230,200), IYAB

C WEDGE COMPRESSION FLOW FIELD
210 CALL COMPR (ANGLE, MER, IPRINT, CPSTAG, ISDET, IFIRST, CP)
WRITE (TAPEOT, 2.5)
215 FORMAT (1H0,46H FLOW FIELD IS BEING GENERATED AND STORED BY ,
1 20HCOMPRESSION ROUTINE. )
GO TO 800

C PRANDTL-MEYER EXPANSION
220 CALL EXPAND (ANGLE, MER, IPRINT, ISDET, CP)
WRITE (TAPEOT, 225)
225 FORMAT (1H0,46H FLOW FIELD IS BEING GENERATED AND STORED BY ,
1 32HPRANDTL-MEYER EXPANSION ROUTINE. )
GO TO 800

C CONICAL FLOW FIELD
230 ANGLF(1) = DINF(5)
CALL CONE (ANGLE, CP, 0)
  
```

FFSPEE

```

WRITE (TAPEOT,235)
235 FORMAT (1H0,46H FLOW FIELD IS BEING GENERATED AND STORED BY ,
1 13HCONE ROUTINE. )
GO TO 800

C
C NEWTONIAN PRANDTL-MEYER FIELD
240 ISF = 0
ETAC = 1.0
IPRCK = 0
CALL NEWTON (ANGLE,EMN,CP,ETAC,IPREK,MEP,CPSTAG,ISE,IFIRST)
WRITE (TAPEOT,245)
245 FORMAT (1H0,46H FLOW FIELD IS BEING GENERATED AND STORED BY ,
1 32HNEWTONIAN PRANDTL-MEYER ROUTINE. )

C
C
C 800 CONTINUE
DINF(1) = BS(6)
DINF(5) = BS(2) / PFS
DINF(6) = BS(3) / TFS

C
C
C STORE DATA
IG10 = LCNEXT
CALL WRITMS (10,DINF,6,IG10)
WRITE (10,IG10) DINF
213 WRITE (TAPEOT,214) TITLE,IMTAB, TITLES,MACH,NAB,LOAD,
1 TITLE,ALPHA,BETA,NREG,LORG, TITLER,IDTYP
214 FORMAT (1H0,7HTITLES,10A4,7H IMTAB,9I4,/1H ,7HTITLES,10A4,
1 7H MACH =,F6.3,5H NAB,13,/1H ,6X,5MLDAB,20I4,/1H ,7HTITLEAB,
2 10A4,7H ALPHA, F6.3,6H BETA, F6.3,6H NREG =,I3,/1H ,4X,5MLORGE,
3 20I4,/1H ,7HTITLES,10A4,7H IDTYPE,5I2)
WRITE (TAPEOT,216) DINF
216 FORMAT (1H0,5HDINF,6F10,2)

C
C SET POINTER TO NEXT EMPTY RECORD NUMBER

```

FFSP 071
FFSP 072
FFSP 073
FFSP 074
FFSP 075
FFSP 076
FFSP 077
FFSP 078
FFSP 079
FFSP 080
FFSP 081
FFSP 082
FFSP 083
FFSP 084
FFSP 085
FFSP 086
FFSP 087
FFSP 088
FFSP 089
FFSP 090
FFSP 091
FFSP 092
FFSP 093
FFSP 094C
FFSP 095I
FFSP 096
FFSP 097
FFSP 098
FFSP 099
FFSP 100
FFSP 101
FFSP 102
FFSP 103
FFSP 104
FFSP 105

FFSPEE

FFSPE

FFSP 106
FFSP 107
FFSP 108
FFSP 109
FFSP 110
FFSP 111
FFSP 112
FFSP 113
FFSP 114

```
C      LCNEXT = IG10 + 1
C      RETURN
C      900 WRITE (TAPEOT,910) ITAB
C      910 FORMAT (1H0,9H**+ITAB Z,12,28H INPUT TO SPECIAL FLOW FIELD,
C      1      53H ROUTINE. IT MUST BE BETWEEN 1 AND 4. PROGRAM STOP, )
C      1      STOP
C      END
```

FFSPE

FFAIRF

```

SURROUTINE FFAIRF
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPFIN, TAPEOT, TAPFA, TAPFS, TAPEC, TAPEO, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEX
COMMON /TAGS/ITAGG, ITAG9, ITAG10
COMMON /FFIELD/TITLEM(10), TITLFS(10), TITLEA(10), TITLER(10),
1 IMTAR(9), IDTYP(5), LOSET(5), LORG(20), L,
2 DINF(6), IRM, LCNEXT, IREG, MACH, NAR, ALPHA, BETA, NREG,
3 NSREG, ITFLAG
C
C DIMENSION TITL(15)
INTEGER ERROR, PAGE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEE, TAPEC, TAPEO, TAPEE, TAPEF,
1 TAPFC, TAPEH, TAPEI, TAPEJ, TAPEX
RETURN
END

```

FFAI 001
FFAI 002
FFAI 003
FFAI 004
FFAI 005
FFAI 006
FFAI 007
FFAI 008
FFAI 009
FFAI 010
FFAI 011
FFAI 012
FFAI 013
FFAI 014
FFAI 015
FFAI 016
FFAI 017

FFAIRF

```

SURROUTINE FPSURF
C
C THIS ROUTINE IS USED TO TRANSFER SURFACE DATA (GENERATED AND
C STORED ON UNIT & BY FORCE) TO THE FLOW FIELD UNIT 10.
C
C
C
COMMON /EXFC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPFIN, TAPFOT, TAPFA, TAPEB, TAPEC, TAPFD, TAPEE, TAPEF,
1 TAPFG, TAPEH, TAPFI, TAPEJ, TAPEK
COMMON /FFIELD/TITLEM(10), TITLFS(10), TITLEA(10), TITLFR(10),
1 IMTAB(9), IDTYP(5), ILOSET(5), LOR(20), LORG(20), L,
2 DINF(6), IRW, ILCNEXT, IREG, MACH, NAB, ALPHA, BETA, NREG,
3 NSREG, ITFLAG
C
DIMENSION TITLE(15), IFC(5), IFD(5), LOSF(5), DAT(6), DATA(12), DATB(25)
DIMENSION IPAHL(10), E(25), EP(25), ELEM(25), E2(25), E1(25)
DIMENSION LOFF(5), LOCO(5)
C
INTEGER ERROR, PAGE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPED, TAPEE, TAPEF,
1 TAPEG, TAPEM, TAPEI, TAPEJ, TAPEK
C
C CHECK READ OR WRITE FLAG
IF (IRW .NE. 0) GO TO 227
C
C READ FLOW FIELD CONTROL AND PLANE ORIENTATION CARD
READ (TAPEIN, 20) NSREG, IFC1, ITFLAG, DAT
20 FORMAT (I1, I4, I1, 4X, 6F10.0)
C
C SET SUB-REGION COUNTERS
IF (NSREG .LE. 0) NSREG = 1
13R = 0

```

FFSU 001
FFSU 002
FFSU 003
FFSU 004
FFSU 005
FFSU 006
FFSU 007
FFSU 008
FFSU 009
FFSU 010
FFSU 011
FFSU 012
FFSU 013
FFSU 014
FFSU 015
FFSU 016
FFSU 017
FFSU 018
FFSU 019
FFSU 020
FFSU 021
FFSU 022
FFSU 023
FFSU 024
FFSU 025
FFSU 026
FFSU 027
FFSU 028
FFSU 029
FFSU 030
FFSU 031
FFSU 032
FFSU 033
FFSU 034
FFSU 035

FFSURF

```

FF8U 036
FF8U 037
FF8U 038C
FF8U 039I
FF8U 040
FF8U 041
FF8U 042
FF8U 043
FF8U 044
FF8U 045
FF8U 046
FF8U 047
FF8U 048
FF8U 049
FF8U 050
FF8U 051
FF8U 052
FF8U 053
FF8U 054
FF8U 055
FF8U 056
FF8U 057
FF8U 058
FF8U 059
FF8U 060
FF8U 061
FF8U 062
FF8U 063
FF8U 064
FF8U 065
FF8U 066
FF8U 067
FF8U 068C
FF8U 069I
FF8U 070

```

FFSURF

```

C      IG10 = LORG(IRFG) + 1
C      CALL WRITMS (10,DAT,6,IG10)
C      WRITE (10,IG10) DAT
C      IG10 = LCNEXT + NSREG - 1

C      25 ISR = ISR + 1
C         LCNEXT = LORG(IREG) + 4 + ISR

C         DO 30 I=1,5
C            LORF(I) = 0
C            LOFF(I) = 0
C            LOCO(I) = 0
C            IFC(I) = 0
C            DATB(I+5) = 0
C            DATE(I+15) = 0
C            IFC(1) = IFC1

C         30 IFC(1) = IFC1

C         LOFF(1) = IG10 + 1
C         IC = 1
C         IDTYP(1) = 2
C         IDTYP(2) = 2

C      C CHECK IF DATA IS TO BE OBTAINED FROM INPUT UNIT OR UNIT 4
C         IF (IFC(1) .EQ. 0) GO TO 39
C      C READ IN DATA FROM INPUT UNIT
C         IFC = IFC(1)
C         DO 36 I=1,IFR
C            READ (TAPEIN,35) DATA
C            35 FORMAT (6F10.0)
C            IG10 = IG10 + 1
C            CALL WRITMS (10,DAT,12,IG10)
C            WRITE (10,IG10) DATA
C         36 CONTINUE

```

FFSURF

FFSU 071
FFSU 072
FFSU 073
FFSU 074
FFSU 075
FFSU 076
FFSU 077
FFSU 078
FFSU 079
FFSU 080
FFSU 081C
FFSU 082I
FFSU 083
FFSU 084
FFSU 085
FFSU 086
FFSU 087
FFSU 088
FFSU 089
FFSU 090
FFSU 091
FFSU 092
FFSU 093
FFSU 094C
FFSU 095I
FFSU 096
FFSU 097
FFSU 098
FFSU 099
FFSU 100
FFSU 101
FFSU 102C
FFSU 103I
FFSU 104
FFSU 105

```
C      K = IFR
C      GO TO 140
C      READ PANEL NUMBERS TO BE GROUPED TO FORM SURFACE COMPONENT
C      39 CONTINUE
C      READ (TAPEIN,40) IPANL
C      40 FORMAT (10I2)
C      OBTAIN POINTER INFO FROM UNIT 4
C      IG4 = 2
C      CALL READMS (4,E,25,IG4)
C      READ (4#IG4) E
C      NRFM = E(4)
C      K = 0
C      START PANEL DO LOOP
C      DO 100 I=1,10
C      IF (IPANL(I) .EQ. 0) GO TO 140
C      WRITE (TAPFOT,50) IPANL(I)
C      50 FORMAT (1H0,49#SURFACE DATA ARE BEING TRANSFERED FROM UNIT 4 TO ,
C      1 17HUNIT 10, IPANL #,13,11H ,2RHDATA ARRAY FOLLOWS FOR EACH ,
C      2 6HPOINT. )
C      IG4 = IPANL(I) * 5
C      CALL READMS (4,EP,25,IG4)
C      READ (4#IG4) EP
C      ISTART = EP(3) -NREM
C      N = EP(4)
C      START ELEMENT DO LOOP WITHIN A PANEL
C      DO 99 II=1,N
C      IG4 = ISTART + NREM*II
C      CALL READMS (4,ELEM,25,IG4)
C      READ (4#IG4) ELEM
C      IG4 = IG4 + 1
```

FFSURF

FFSURF

FFSU 106
 FFSU 107C
 FFSU 10MI
 FFSU 109
 FFSU 110
 FFSU 111C
 FFSU 112I
 FFSU 113
 FFSU 114
 FFSU 115
 FFSU 116
 FFSU 117
 FFSU 118
 FFSU 119
 FFSU 120
 FFSU 121
 FFSU 122
 FFSU 123
 FFSU 124
 FFSU 125
 FFSU 126
 FFSU 127
 FFSU 128
 FFSU 129
 FFSU 130
 FFSU 131C
 FFSU 132I
 FFSU 133
 FFSU 134
 FFSU 135
 FFSU 136
 FFSU 137
 FFSU 13A
 FFSU 139
 FFSU 140

FFSURF

```

C      CALL READMS (4,E2,25,IG41)
C      READ (4*IG41) E2
C      IG4S = E2(NAR)

C      CALL READMS (4,E1,25,IG4S)
C      READ (4*IG4S) E1

C      DATA(1) = ELEM(7)
C      DATA(2) = ELEM(8)
C      DATA(3) = ELEM(9)
C      DATA(4) = 0.0
C      DATA(5) = 0.0
C      DATA(6) = 0.0
C      DATA(7) = E1(7)
C      DATA(8) = E1(8)
C      DATA(9) = E1(9)
C      DATA(10) = E1(10)
C      DATA(11) = E1(11)
C      DATA(12) = E1(12)
C      WRITE (TAPEOT,52) I1,IG4S,DATA
C      52  FORMAT (1H ,2I5,3X,6E12.5,/1H ,13X,6E12.5)

C      IG10 = IG10 + 1

C      CALL WRITMS (10,DATA,12,IG10)
C      WRITE (10*IG10) DATA
C      K = K + 1

C      99 CONTINUE
C      100 CONTINUE
C      140 CONTINUE
C      IFC(1) = K

```

FFSURF

```

C SAVE LOCATIONS AND COUNTERS FOR SUB-REGION
DO 260 I = 1,5
  DATB(I) = LOFF(I)
  DATB(I+5) = JFC(I)
260 DATB(I+10) = LOCD(I)
C
C CALL WRITMS (10, DATB,25,LCNEXT)
C WRITE(10=LCNEXT) DATB
C LCNEXT = IGI0 + 1
C IF (ISR .LT. NSREG) GO TO 25
C GO TO 600

C FLOW FIELD DATA WILL BE READ FROM UNIT 10
227 WRITE (TAPECT,217)
217 FORMAT (1H0,48MSURFACE DATA ARE BEING READ FROM FLOW-FIELD UNIT )
  WRITE (TAPECT,214) TITLE,IMTAB, TITLES,MACH,LOADP,
    1 TITLE,ALPHA,BETA,NREG,LORG, TITLER,DTYP
214 FORMAT (1H0,7HTITLEM,10A4,7H IMTAB,9I4,/1H ,7HTITLES,10A4,
  1 7H MACH ,F6.3,5H MAB,13,/1H ,4X,5HLOADP,20I4,/1H ,7HTITLEA,
  2 10A4,7H ALPHA,F6.3,6H BETA,F6.3,6H NREG ,F6.3,13,/1H ,4X,5HLOADP,
  3 20I4,/1H ,7HTITLES,10A4,7H DTYP,5I2)
  IGI0 = LORG(ITREG) + 1
  CALL READMS(10, DAT, 6, IGI0)
  READ (10=IGI0) DAT
  WRITE(TAPECT,228) DAT
228 FORMAT(1H0,
  2 3HXO,F7.3,3X,3HYO,F7.3,3X,3HZO,F7.3,
  3 3X,5HPSIO,F7.3,3X,6HTHETO,F7.3,6X,5HPHIO,F7.3)
C
C READ TN SUB-REGIONS
DO 800 ISR = 1,NSREG
  IGI0 = LORG(ITREG) + 4+ ISR
  ISF = 0

```

```

FFSU 141
FFSU 142
FFSU 143
FFSU 144
FFSU 145
FFSU 146
FFSU 147
FFSU 148C
FFSU 149I
FFSU 150
FFSU 151
FFSU 152
FFSU 153
FFSU 154
FFSU 155
FFSU 156
FFSU 157
FFSU 158
FFSU 159
FFSU 160
FFSU 161
FFSU 162
FFSU 163
FFSU 164
FFSU 165C
FFSU 166I
FFSU 167
FFSU 168
FFSU 169
FFSU 170
FFSU 171
FFSU 172
FFSU 173
FFSU 174
FFSU 175

```

FFSURF

FFSU 176
 FFSU 177
 FFSU 17AC
 FFSU 179I
 FFSU 180
 FFSU 181
 FFSU 182
 FFSU 183
 FFSU 184
 FFSU 185
 FFSU 186
 FFSU 187
 FFSU 188
 FFSU 189
 FFSU 190
 FFSU 191
 FFSU 192
 FFSU 193
 FFSU 194
 FFSU 195
 FFSU 196
 FFSU 197
 FFSU 198
 FFSU 199
 FFSU 200
 FFSU 201
 FFSU 202
 FFSU 203
 FFSU 204
 FFSU 205
 FFSU 206
 FFSU 207
 FFSU 208
 FFSU 209
 FFSU 210

```

700 CONTINUE
C   CALL READMS (10, DATB,25,1610)
C   READ(10,IG10) DATB
  DO 710 I = 1,5
    LOFF(I) = DATB(I) + 0.001
    IFC(I) = DATB(I+5) + 0.001
    LOCD(I) = DATB(I+10) + 0.001
    IFD(I) = DATB(I+15) + 0.001
  710 CONTINUE
C
C   COUNT NUMBER OF SECONDARY FLOWS (SUB-REGIONS ONLY)
  IF (ISF,NE,0) GO TO 750
  NSECF = 0
  DO 720 I = 1,5
    LOSF(I) = DATB(1+20) + 0.001
  720 CONTINUE
  730 CONTINUE
  WRITE(TAPEOT,740) IREG, ISR, NSECF
  740 FORMAT(1H1, 11HFLOW REGION, I3/1H0, 10HSUB-REGION, I3,
    1 20X 25HNUMBER OF SECONDARY FLOWS, I3)
C
  750 WRITE(TAPEOT,760) LOFF, IFC, LOCD, IFD
  760 FORMAT(1H0, 4HLOFF, 5I20/1H, 4H IFC, 5I20/
    1 1H0, 4HLOCD, 5I20/1H, 4H IFD, 5I20)
C
  IG10 = LOFF(1) - 1
  IFR = IFC(1)
  IC = 1
  231 WRITE (TAPEOT,237) IC
  237 FORMAT (1H0,9HBOUNDARY ,I2,5H DATA )
  236 DO 232 I=1,IFR

```

```

IG10 = IG10 + 1
CALL READM8 (10,DATA,12,IG10)
READ (10,IG10) DATA
232 WRITE (TAPEOT,233) DATA
233 FORMAT (1H , 12F10.4)
IF (IC .NE. 1) GO TO 234
IC = 2
IFR = IFC(2)
IF (IFR .EQ. 0) GO TO 234
GO TO 231
234 IF (IC .NE. 2) GO TO 235
IC = 3
IFR = IFC(3)
IF (IFR .EQ. 0) GO TO 235
GO TO 231
235 IF (IC .NE. 3) GO TO 239
IC = 4
IFR = IFC(4)
IF (IFR .EQ. 0) GO TO 239
GO TO 231
239 IF (IC .NE. 4) GO TO 245
IC = 5
IFR = IFC(5)
IF (IFR .EQ. 0) GO TO 245
WRITE (TAPEOT,238)
238 FORMAT (1H0,25HREMAINING FLOW-FIELD DATA )
GO TO 236
C 245 CONTINUE
C
C CHECK FOR SECONDARY FLOWS
IF (ISF .EQ. NSECF) GO TO 800
ISF = ISF + 1
WRITE(TAPEOT,776) ISF
770 FORMAT(1H1, 14HSECONDARY FLOW, I3)

```

FFSU 211
FFSU 2120
FFSU 2131
FFSU 214
FFSU 215
FFSU 216
FFSU 217
FFSU 218
FFSU 219
FFSU 220
FFSU 221
FFSU 222
FFSU 223
FFSU 224
FFSU 225
FFSU 226
FFSU 227
FFSU 228
FFSU 229
FFSU 230
FFSU 231
FFSU 232
FFSU 233
FFSU 234
FFSU 235
FFSU 236
FFSU 237
FFSU 238
FFSU 239
FFSU 240
FFSU 241
FFSU 242
FFSU 243
FFSU 244
FFSU 245

FFSURF

```
C C SET LOCATIONS
  I610 = LOSP(ISF)
  GO TO 700
C A00 CONTINUE
C 600 CONTINUE
C
C RETURN
C END
```

```
FFSU 246
FFSU 247
FFSU 248
FFSU 249
FFSU 250
FFSU 251
FFSU 252
FFSU 253
FFSU 254
FFSU 255
FFSU 256
FFSU 257
FFSU 258
```

FFSURF

FFINPT

```

C          CURROUTINE FFINPT
C THIS ROUTINE IS USED TO PERMIT HAND LOADING OF FLOW FIELD DATA ON
C UNIT 10.
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOUT, TAPFA, TAPFB, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPFK
COMMON /TAGS/ITAG4, ITAG9, ITAG10
COMMON /FFIELD/TITLEM(10), TITLES(10), TITLEA(10), TITLER(10),
1 IMTAR(9), IDTYP(5), LOSET(5), LOAR(20), LORG(20), L,
2 CINF(6), IRM, LCNEXT, IREG, MACH, NAB, ALPHA, BETA, NREG,
3 NSREG, ITFLAG
C
DIMENSION TITLF(15), IFC(5), IFD(5), LOSF(5), DAT(6), DATA(12), DATB(25)
DIMENSION LOFF(5), LOCD(5)
C
INTEGER ERROR, PAGE
INTEGER TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
C
READ (TAPEIN, 10) IDTYP(2)
10 FORMAT (I1)
ITAB = IDTYP(2)
IF (ITAB.LT.1 .OR. ITAB.GT.4) GO TO 900
GO TO (210, 220, 230, 240), ITAB
C
C UNIFORM FLOW FIELD - SAME FOR ALL X, Y, Z.
210 IF (IRM .NE. 0) GO TO 212
READ (TAPFIN, 211) DINF
211 FORMAT (C 10, 0)
IG10 = LCNEXT
CALL WRITMS (10, DINF, 6, IG10)
C
WRITE (10, IG10) DINF

```

FFIN 001
FFIN 002
FFIN 003
FFIN 004
FFIN 005
FFIN 006
FFIN 007
FFIN 008
FFIN 009
FFIN 010
FFIN 011
FFIN 012
FFIN 013
FFIN 014
FFIN 015
FFIN 016
FFIN 017
FFIN 018
FFIN 019
FFIN 020
FFIN 021
FFIN 022
FFIN 023
FFIN 024
FFIN 025
FFIN 026
FFIN 027
FFIN 028
FFIN 029
FFIN 030
FFIN 031
FFIN 032
FFIN 033
FFIN 034C
FFIN 035I

FFINPT

FFINPT

```

C SET POINTER TO START OF DATA
  LOFF(1) = LCNEXT
  LCNEXT = IG10 + 1
  WRITE (TAPEOT,216)
216 FORMAT (1H0,43HDATA ARE BEING READ INTO UNIFORM FLOW FIELD )
  GO TO 213
217 IG10 = LCNEXT
  WRITE (TAPEOT,217)
217 FORMAT (1H0,40HDATA ARE BEING READ FROM FLOW-FIELD UNIT)
  CALL READMS (10,DINF,6,IG10)
  READ (10,IG10) DINF
C
213 WRITE (TAPEOT,218) TTYLEM,IMTAR, TTILES,MACH,NAB9,LOAD,
  1 TTILEA,ALPHA,BETA,NREG,LORG, TITLER, IDTYP
214 FORMAT (1H0,7HTTYLEM=,10A4,7H IMTAR=,9T0,/1H ,7HTTYLES=,10A4,
  1 7H MACH =,F6.3,5H NAB=,13,/1H ,4X,5HLCAB=,20I4,/1H ,7HTTILEAS,
  2 10A4,7H ALPHA=,F6.3,6H BETA=,F6.3,8H NREG =,I3,/1H ,4X,5HLCOR=,
  3 20I4,/1H ,7HTTILERS=,10A4,7H IDTYP=,5I2)
  WRITE (TAPEOT,215) DINF
215 FORMAT (1H0,5HDINF=,6F10.2)
  GO TO 600
C
C
C
C NON-UNIFORM FLOW-FIELD DATA ARE INPUT UP READ
220 IF (TRW .NE. 0) GO TO 227
C
C READ ORIENTATION CARD
  READ(TAPEIN,221) NSREG, ITFLAG, DAT
221 FORMAT(2(I1,4X), 6F10.0)
C SFT SUB-REGION COUNTERS
  IF (NSREG .LF. 0) NSREG = 1
  ISR = 0
C
  IG10 = LORG(IREG) + 1
  CALL WRITMS (10, DAT, 6, IG10)

```

FFINPT

FFINPT

```
C WRITE(10#IG10) DAY
  IG10 = LCNEXT + NSREG + 1
  100 ISR = ISR + 1
  LCNEXT = LORC(IREG) + 4 + ISR
  ISF = 0
  DO 249 I = 1,5
  249 LORF(I) = 0
  110 CONTINUE

C READ IN FLOW FIELD CONTROL CARD
  READ(TAPEIN,229) IFC, IFD, ISECF
  229 FORMAT(11I2)

C PRESET LOCATIONS AND COUNTERS
  DO 250 I = 1,5
  LORF(I) = 0
  LORC(I) = 0
  DATR(I+5) = IFC(I)
  CATR(I+15) = IFD(I)
  250 CONTINUE

C
  LORF(1) = IG10 + 1
  IFR = IFC(1)
  IC = 1

C READ IN FLOW FIELD DATA. IC=1 BOUNDARY 1; IC=2 BOUNDARY 2;
C IC=3 BOUNDARY 3; IC=4 BOUNDARY 4; IC=5 REMAINING FLOW FIELD DATA
  222 DO 224 I=1,IFR
  READ (TAPEIN,223) DATA
  223 FORMAT (6F10.0)
  IG10 = IG10 + 1
  224 CALL WRITMS (10,DATA,12,IG10)
C 224 WRITE (10#IG10) DATA
C IF (IC .NE. 1) GO TO 225
  IC = 2
  LORF(2) = IG10 + 1
```

FFIN 0711
FFIN 072
FFIN 073
FFIN 074
FFIN 075
FFIN 076
FFIN 077
FFIN 078
FFIN 079
FFIN 081
FFIN 082
FFIN 083
FFIN 084
FFIN 085
FFIN 086
FFIN 087
FFIN 088
FFIN 089
FFIN 090
FFIN 091
FFIN 092
FFIN 093
FFIN 094
FFIN 095
FFIN 096
FFIN 097
FFIN 098
FFIN 099
FFIN 100
FFIN 101C
FFIN 1021
FFIN 103
FFIN 104
FFIN 105

FFINPT

FFINPT

PFIN 106
 PFIN 107
 PFIN 108
 PFIN 109
 PFIN 110
 PFIN 111
 PFIN 112
 PFIN 113
 PFIN 114
 PFIN 115
 PFIN 116
 PFIN 117
 PFIN 118
 PFIN 119
 PFIN 120
 PFIN 121
 PFIN 122
 PFIN 123
 PFIN 124
 PFIN 125
 PFIN 126
 PFIN 127
 PFIN 128
 PFIN 129
 PFIN 130
 PFIN 131
 PFIN 132
 PFIN 133
 PFIN 134C
 PFIN 135I
 PFIN 136
 PFIN 137
 PFIN 138
 PFIN 139
 PFIN 140

```

IFR = IFC(2)
IF (IFR .EQ. 0) GO TO 225
GO TO 222
225 IF (IC .NE. 2) GO TO 226
IC = 3
IFR = IFC(3)
IF (IFR .EQ. 0) GO TO 226
LOFF(3) = IG10 + 1
GO TO 222
226 IF (IC .NE. 3) GO TO 241
IC = 4
IFR = IFC(4)
IF (IFR .EQ. 0) GO TO 241
LOFF(4) = IG10 + 1
GO TO 222
241 IF (IC .NE. 4) GO TO 242
IC = 5
IFR = IFC(5)
IF (IFR .EQ. 0) GO TO 242
LOFF(5) = IG10 + 1
GO TO 222
242 CONTINUE
C SAVE LOCATIONS AND COUNTERS FOR SUB-REGION
DO 260 I = 1,5
  DATB(I) = LOFF(I)
  260 DATR(I+10) = LOCD(I)
C CALL WRITMS (10, DATB,25,LCNEXT)
C WRITE(10#LCNEXT) DATB
C CHECK FOR SECONDARY FLOW
  IF (ISECF .EQ. 0) GO TO 300
C SECONDARY FLOW
  
```

FFINPT

FFINPT

FFIN 141
FFIN 142
FFIN 143
FFIN 144
FFIN 145
FFIN 146
FFIN 147
FFIN 148
FFIN 149
FFIN 150
FFIN 151
FFIN 152
FFIN 153
FFIN 154C
FFIN 155I
FFIN 156
FFIN 157
FFIN 158
FFIN 159C
FFIN 160I
FFIN 161
FFIN 162
FFIN 163
FFIN 164
FFIN 165
FFIN 166
FFIN 167
FFIN 168
FFIN 169
FFIN 170
FFIN 171
FFIN 172
FFIN 173
FFIN 174
FFIN 175C

```
IG10 = IG10 + 1
LCNEXT = IG10
ISF = ISF + 1
LOSF(ISF) = LCNEXT
GO TO 110

C 300 CONTINUE
  IF (ISF .EQ. 0) GO TO 500
C RESET LOCATIONS FOR SECONDARY FLOWS
  LCNEXT = LOPC(IREG) + 4 + ISR
C CALL READMS (10, DATB,25,LCNEXT)
  READ(10,LCNEXT) DATB
  DO 310 J = 1,ISF
C 310 DATA(I+20) = LOSF(I)
C CALL WRITMS (10, DATA,25,LCNEXT)
  WRITE(10,LCNEXT) DATA
C 500 CONTINUE
  IF (TSR .LT. NSPEG) GO TO 100
  LCNEXT = IG10 + 1
  GO TO 400
C FLOW FIFD DATA WILL BE READ FROM UNIT 10
227 WRITE (TAPE07,217)
  WRITE (TAPE07,214) TITLE,IMTAB, TITLES,NACH,NAB,LOAD,
  1 TITLE,ALPHA,BETA,NREG,LORG, TITLER,YD1YP
  IG10 = LORG(IREG) + 1
  CALL READMS(10, DAT, 6, IG10)
```

FFINPT

FFINPT

```

C   READ (10*IG10) DAT
    WRITE(TAPEOT,228) DAT
228 FORMAT(1H0,
2   3HX0=F7.3,3X,3HY0=F7.3,3X,3HZ0=F7.3,
3   3X,5HPSI0=F7.3,3X,6HTHETO=F7.3,6X,5HPHIO=F7.3)

C   READ IN SUB-REGIONS
C   DO 800 ISR = 1,NSREG
    IG10 = LORG(IREG) + 4 + ISR
    ISF = 0
700 CONTINUE

C   CALL READMS (10, DAT8,25,IG10)
C   READ(10*IG10) DAT8
    DO 710 I = 1,5
        LOFF(I) = DAT8(I)      + 0.001
        IF (I) = DAT8(I+5)    + 0.001
        LLOC(I) = DAT8(I+10)  + 0.001
        IFD(I) = DAT8(I+15)  + 0.001
710 CONTINUE

C
C   COUNT NUMBER OF SECONDARY FLOWS (SUB-REGIONS ONLY)
    IF (ISF.NE. 0) GO TO 750
    NSECF = 0
    DO 720 I = 1,5
        LOSF(I) = DAT8(I+20) + 0.001
        IF (LOSF(I).EQ. 0) GO TO 730
        NSECF = NSECF + 1
720 CONTINUE
730 CONTINUE
    WRITE(TAPEOT,740) IREG, ISR, NSECF
740 FORMAT(1H1, 11HFLOW REGION, I3/1H0, 10HNSUP-REGION, I3,
1   20X 25HNUMBER OF SECONDARY FLOWS, I3)
C

```

FFINBT

FFINPT

```

750 WRITE(TAPEOT,760) LOFF, IFC, LOCD, IFD
760 FORMAT(1H0, 4HLOFF, 5I20/1H , 4H IFC, 5I20/
      1 1H0, 4HLOCD, 5I20/1H , 4H IFD, 5I20)

```

C

```

      IG10 = LOFF(1) * 1
      IFR = IFC(1)
      IC = 1
231 WRITE (TAPEOT,237) IC
237 FORMAT (1H0,9HBOUNDARY ,12,5H DATA )
236 DO 232 I=1,IFR

```

```

      IG10 = IG10 + 1
      CALL READMS (10,DATA,12,IG10)
      READ (10,IG10) DATA
232 WRITE (TAPEOT,233) DATA
233 FORMAT (1H , 12F10,4)
      IF (IC .NE. 1) GO TO 234

```

C

```

      IC = 2
      IFR = IFC(2)
      IF (IFR .NE. 0) GO TO 231
234 IF (IC .NE. 2) GO TO 235

```

```

      IC = 3
      IFR = IFC(3)
      IF (IFR .NE. 0) GO TO 231
235 IF (IC .NE. 3) GO TO 239

```

```

      IC = 4
      IFR = IFC(4)
      IF (IFR .NE. 0) GO TO 231
239 IF (IC .NE. 4) GO TO 245

```

```

      IC = 5
      IFR = IFC(5)
      IF (IFR .EQ. 0) GO TO 245
      WRITE (TAPEOT,238)

```

```

238 FORMAT (1H0,25HREMAINING FLOW=FIELD DATA )
      GO TO 236

```

C

```

FFIN 211
FFIN 212
FFIN 213
FFIN 214
FFIN 215
FFIN 216
FFIN 217
FFIN 218
FFIN 219
FFIN 220
FFIN 221
FFIN 222C
FFIN 223I
FFIN 224
FFIN 225
FFIN 226
FFIN 227
FFIN 228
FFIN 229
FFIN 230
FFIN 231
FFIN 232
FFIN 233
FFIN 234
FFIN 235
FFIN 236
FFIN 237
FFIN 238
FFIN 239
FFIN 240
FFIN 241
FFIN 242
FFIN 243
FFIN 244
FFIN 245

```

FFINPT

FFINPT

FFIN	246
FFIN	247
FFIN	248
FFIN	249
FFIN	250
FFIN	251
FFIN	252
FFIN	253
FFIN	254
FFIN	255
FFIN	256
FFIN	257
FFIN	258
FFIN	259
FFIN	260
FFIN	261
FFIN	262
FFIN	263
FFIN	264
FFIN	265
FFIN	266
FFIN	267
FFIN	268
FFIN	269
FFIN	270
FFIN	271

FFINPT

```

245 CONTINUE
C
C CHECK FOR SECONDARY FLOWS
IF (.EQ. NSECF) GO TO 800
ISF = ISF + 1
WRITE(TAPEOT,770) ISF
770 FORMAT(1H1, 14HSECONDARY FLOW, 13)
C
C SET LOCATIONS
IG10 = LOSF(ISF)
GO TO 700
C
800 CONTINUE
C
230 CONTINUE
GO TO 600
240 CONTINUE
600 CONTINUE
C
C
C
RETURN
900 WRITE (TAPEOT,910)
910 FORMAT (1H ,47H** ITAB PARAMETER INPUT TO FFINPT IS WRONG*STOP )
STOP
END

```

FFBODY

```

SUBROUTINE FF600Y
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPFK
1 COMMON /FFIELD/TITLEM(10), TITLES(10), TITLEA(10), TITLER(10),
      INTAR(9), IDTYP(5), ILOSET(5), LQAR(20), LORG(20), L,
2      DINF(6), IRM, LCNEXT, IRFG, MACH, NAB, ALPHA, BETA, NREG,
3      NSREG, ITFLAG
COMMON/SOSEIN/ AMF, ALFWD, PHIT, G, IFLG(5), NP, AMP, ALFP,
      A(100), R(100)
1 COMMON/SOSFOT/AMANS(100), AMDX(100), AMDY(100), AMDZ(100),
      PANS(100), TD(100), AMS(100), AMSX(100), AMSY(100), AMSZ(100),
2      PSMK(100), TS(100), NSP, NS, JS(10), IS(10), LS(10), IN(10),
3      XANS(100), YANS(100), XS(100), YS(100)
C
COMMON /GASD/GAM, GASCP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
DIMENSION TITL(15), NE(10), DAT(6)
DIMENSION COMPID(10), PHI(40), YPA(40), DATA(12), LOSF(5)
REAL MACH
REAL LOC(25)
C
INTEGER ERROP, PAGE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
C
DATA RC /1.705329252E-2/
C
TFX(XO, X, Y, Z) = XO + AP11*X + AP21*Y + AP31*Z
C
TFY(YO, X, Y, Z) = YO + AP12*X + AP22*Y + AP32*Z
C
TFZ(ZO, X, Y, Z) = ZO + AP13*X + AP23*Y + AP33*Z
C
G = GAM
C

```

FFBODY

FFBODY

```

C READ IN SHOCK EXPANSION FLOW FIELD CONTROL CARD
  READ(TAPEIN, 10) IFLG, X1, X2
  10 FORMAT( 5I1, 5X, 2F10.0)

C GO GET MERIDIAN CUTS, RAW DATA (Y TAPEB
  AND BY MERIDIAN ON TAPEC,
C IF IFLG(2) = 1, USE PREVIOUSLY CALCULATED DATA
  YF (IFLG(2) .EQ. 1) GO TO 20
  CALL MERID

C NOW ORDR MERIDIAN DATA. SAVE FINAL RESULTS ON TAPEB.
  CALL MOUT

C 20 REWIND TAPEB
  READ(TAPEB)
  1
  2
  COMPID, NC, NE, NPL, MPTOT,
  PSIO, THED, PHIO, XPO, YPO, ZPO, IYTP1,
  (PHI(I), I=1, NPL), (YPA(I), I=1, NPL)

C NOW READY TO CALCULATE FLOW FIELD DATA
  USING SHOCK EXPANSION METHOD.

C PREPARE VELOCITY AND TRANSFORMATION MATRICES.

  ALPHA = RC*ALPHA
  BETAR = RC*BETA
  COSA = COS(ALPHAR)
  SINA = SIN(ALPHAR)
  COSR = COS(BETAR)
  SINR = SIN(BETAR)
  COSPS = COS(RC*PSIO)
  SINPS = SIN(RC*PSIO)
  COST = COS(RC*THEN)
  SINT = SIN(RC*THEN)

C DIRECTION COSINES OF FREE STREAM FLOW

```

FFBODY

FFBODY

```
AMF = MACH
AMFX = -COSA*COSE
AMFY = SINB
AMFZ = SINA*COSE

C CALCULATE DIRECTION COSINES OF FREE STREAM
C IN MERIDIAN AXIS COORDS.
C
C COMPONENTS OF TRANSFORMATION MATRIX
AA11 = COST*COSE
AA12 = COST*SINPS
AA13 = -SINT
C
AA21 = -SINPS
AA22 = COSE
AA23 = 0.0
C
AA31 = -SINT*COSE
AA32 = -SINT*SINPS
AA33 = -COST
C
C DIRECTION COSINES IN MERIDIAN AXIS COORDS.
VX = AA11*AMFX + AA12*AMFY + AA13*AMFZ
VY = AA21*AMFX + AA22*AMFY + AA23*AMFZ
VZ = AA31*AMFX + AA32*AMFY + AA33*AMFZ
C
C CALCULATE LOCATION OF WINDWARD PLANE
PHTW = 0.0
IF (VY .EQ. 0.0 .AND. VZ .LE. 0.0) GO TO 30
PHTW = 3.1415926536
IF (VY .EQ. 0.0) GO TO 30
PHTW = ATAN2(VY,VZ)
30 CONTINUE
C
C CALCULATE ANGLE OF ATTACK IN WINDWARD PLANE
```

```
FF8C 071
FF8C 072
FF8C 073
FF8C 074
FF8C 075
FF8C 076
FF8C 077
FF8C 078
FF8C 079
FF8C 080
FF8C 081
FF8C 082
FF8C 083
FF8C 084
FF8C 085
FF8C 086
FF8C 087
FF8C 08C
FF8C 089
FF8C 090
FF8C 091
FF8C 092
FF8C 093
FF8C 094
FF8C 095
FF8C 096
FF8C 097
FF8C 098
FF8C 099
FF8C 100
FF8C 101
FF8C 102
FF8C 103
FF8C 104
FF8C 105
```

FFBODY

FF80DY

FF80	106
FF80	107
FF80	108
FF80	109
FF80	110
FF80	111
FF80	112
FF80	113
FF80	114
FF80	115
FF80	116
FF80	117
FF80	118
FF80	119
FF80	120
FF80	121
FF80	122
FF80	123
FF80	124
FF80	125
FF80	126
FF80	127
FF80	128
FF80	129
FF80	130
FF80	131
FF80	132
FF80	133
FF80	134
FF80	135
FF80	136
FF80	137
FF80	138
FF80	139
FF80	140

FF80DY

```

C ALFWD = ACOS(-VX)
C BEGIN CYCLE ON SPECIFIC PLANES (NPL OF THFM)
C THIS IS CYCLE ON SUB-REGIONS
  NSREG = NPL
  LCNEXT = LCNEXT + NSREG
  DO 1500 N = 1, NPL
C TRANSFORMATION MATRIX FOR GIVEN PLANE (PHI + YPA).
  AP1 = AA11
  AP12 = AA12
  AP13 = AA13
  Cosp = COS(PHI(N))
  SInP = SIN(PHI(N))
  AP21 = Cosp*AA21 - SInP*AA31
  AP22 = Cosp*AA22 - SInP*AA32
  AP23 = -SInP*AA33
  AP31 = SInP*AA21 + Cosp*AA31
  AP32 = SInP*AA22 + Cosp*AA32
  AP33 = Cosp*AA33
  XO = XPO
  YO = YPO + YPA(N)
  ZO = ZPO
C LOCATION FROM WINDWARD PLANE
  PHIT = (PHI(N) + PHIW)
  MACH NO. IN PHIT PLANE, AMP.
  EMC = VZ+COS(PHIT) + VY*SIN(PHIT)
  AMP = AMF*SQRT(VX**2 + EMC**2)
C ANGLE OF AMP TO AXIS
  ALFP = ATAN2(-F4C, VYX)

```

FFBODY

```
C READ IN MERIDIAN DATA
  READ(TAPEB) NP
  DO 90 J = 1, NP
  READ(TAPFB) IC, COMID, JJ, XI, YI, ZI, R(J), A(J)
90 CONTINUE

C
C CALL SOSE

C SET UP FLOW FIELD REGIONS (NS OF THEM),
C
C CYCLE ON SECONDARY FLOWS
  IG10 = LORG(IREG) + A + N
  NSFCF = NS - 1
  DO 95 K = 10, 25
95 LOC(K) = 0

C
C ISF = 0
  IR = 0

C GATHER, TRANSFORM AND STORE FLOW FIELD DATA,
100 IR = IR + 1
  ITAG10 = LCNEXT
  DO 105 K = 1, 9
105 LOC(K) = 0

C SET COUNTERS FOR BOUNDARY 1
  LCC(1) = LCNEXT
  I1 = IS (IR)
  I2 = NP
  IF (NS .GT. IR) I2 = IS(IR + 1) - 1
  IL1 = 0
  DO 107 I = I1, I2
  IF (IL1 .NE. 0) GO TO 106
  IF (XI .GE. XANS(I)) IL1 = ?
```

FFBODY

FF90 141
FF80 142
FF80 143
FF90 144
FF90 145
FF80 146
FF80 147
FF80 148
FF80 149
FF80 150
FF80 151
FF80 152
FF90 153
FF80 154
FF80 155
FF80 156
FF80 157
FF80 158
FF80 159
FF80 160
FF80 161
FF80 162
FF90 163
FF80 164
FF80 165
FF80 166
FF80 167
FF80 168
FF90 169
FF80 170
FF80 171
FF90 172
FF80 173
FF80 174
FF80 175

FFBODY

```

106 IF (X2 .GE. XANS(I)) GO TO 108
107 CONTINUE
    IL2 = I2
    GO TO 109
108 IL2 = I
109 IF (IL1 .LT. I1) IL1 = I1
    I1 = IL1
    I2 = IL2
    LOC(6) = I2 = I1 + 1
    I8 = 1
    GO TO 200

C SET COUNTERS FOR BOUNDARY 2
110 I1 = JS(IR)
    I2 = LS(IR)
    IL3 = 0
    DO 117 I = I1, I2
    IF (IL3 .NE. 0) GO TO 116
    IF (X1 .GE. XS(I)) I13 = I
    IF (X2 .GE. XS(I)) GO TO 118
117 CONTINUE
    IL4 = I2
    GO TO 119
118 IL4 = I
119 IF (I13 .LT. I1) IL3 = I1
    I1 = IL3
    I2 = IL4
    I4 = 1
    LOC(2) = LOC(1) + LOC(6)
    LOC(7) = I2 = I1 + 1
    GO TO 300

C SET COUNTERS FOR BOUNDARY 3
120 I1 = IL1
    I2 = I1

```

```

FF80 176
FF80 177
FF80 178
FF80 179
FF80 180
FF80 181
FF80 182
FF80 183
FF80 184
FF80 185
FF80 186
FF80 187
FF80 188
FF80 189
FF80 190
FF80 191
FF80 192
FF80 193
FF80 194
FF80 195
FF80 196
FF80 197
FF80 198
FF80 199
FF80 200
FF80 201
FF80 202
FF80 203
FF80 204
FF80 205
FF80 206
FF80 207
FF80 208
FF80 209
FF80 210

```

FFBODY

FF80 211
 FF80 212
 FF80 213
 FF80 214
 FF80 215
 FF80 216
 FF80 217
 FF80 218
 FF80 219
 FF80 220
 FF80 221
 FF80 222
 FF80 223
 FF80 224
 FF80 225
 FF80 226
 FF80 227
 FF80 228
 FF80 229
 FF80 230
 FF80 231
 FF80 232
 FF80 233
 FF80 234
 FF80 235
 FF80 236
 FF80 237
 FF80 238
 FF80 239
 FF80 240
 FF80 241
 FF80 242
 FF80 243
 FF80 244
 FF80 245

```

  IR = 2
  GO TO 200
130 I1 = IL3
  I2 = I1
  IK = 2
  LOC(3) = LOC(2) + LOC(7)
  LOC(8) = 2
  GO TO 300

C SET COUNTERS FOR BOUNDARY 4
140 I1 = IL2
  I2 = I1
  IR = 3
  LOC(4) = LOC(3) + LOC(8)
C THE FOLLOWING CARD IS A DUMMY TEST UNTIL INTERSECTIONS
C BECOME ACTIVE. CARD SHOULD BE
C IF (NS .GT. 14) GO TO 160
C IF (NS .EQ. 0) GO TO 160
  GO TO 200
150 I1 = IL4
  I2 = I1
  IK = 4
  LOC(9) = 2
  GO TO 300
160 I1 = JS(IR+1)
  I2 = IN(IR)
  IK = 3
  LOC(9) = I2 - I1 + 1
  GO TO 300
170 I1 = LS(IR)
  I2 = I1
  IK = 4
  LOC(9) = LOC(9) + 1
  GO TO 300
C

```

FFBODY

FF80 246
FF80 247
FF80 248
FF80 249
FF80 250
FF80 251
FF80 252
FF80 253
FF80 254
FF80 255
FF80 256
FF80 257
FF80 258
FF80 259
FF80 260
FF80 261
FF80 262C
FF80 263I
FF80 264
FF80 265
FF80 266
FF80 267
FF80 268
FF80 269
FF80 270
FF80 271
FF80 272
FF80 273
FF80 274
FF80 275
FF80 276
FF80 277
FF80 278
FF80 279
FF80 280

```
C 200 CONTINUE
   DO 210 I = 11,12
   DATA(1) = TFX(XD, XANS(I), 0.0, YANS(I))
   DATA(2) = TFY(YD, XANS(I), 0.0, YANS(I))
   DATA(3) = TFZ(ZD, XANS(I), 0.0, YANS(I))
   DATA(4) = XANS(I)
   DATA(5) = YANS(I)
   DATA(6) = PHI(N)
   DATA(7) = AMANS(I)
   DATA(8) = AMDX(I)
   DATA(9) = AMDZ(I)
   DATA(10) = AMDY(I)
   DATA(11) = PANS(I)
   DATA(12) = TD(I)

C CALL WRITMS(10, DATA, 12, ITAG10)
C WRITE(10,ITAG10) DATA
C ITAG10 = ITAG10 + 1

C 210 CONTINUE
   GO TO (110, 130, 150), IB

C 300 CONTINUE
   DO 310 I = 11,12
   DATA(1) = TFX(XD, XS(I), 0.0, YS(I))
   DATA(2) = TFY(YD, XS(I), 0.0, YS(I))
   DATA(3) = TFZ(ZD, XS(I), 0.0, YS(I))
   DATA(4) = XS(I)
   DATA(5) = YS(I)
   DATA(6) = PHI(N)
   DATA(7) = AMS(I)
   DATA(8) = AMSX(I)
   DATA(9) = AMSZ(I)
```

FFBODY

FF80DY

FF80	261
FF80	262
FF80	263
FF80	264
FF80	265C
FF80	261J
FF80	287
FF80	28A
FF80	2A9
FF80	290
FF80	291
FF80	292
FF80	293
FF80	294C
FF80	295I
FF80	296
FF80	297
FF80	298
FF80	299
FF80	300
FF80	301
FF80	302
FF80	303
FF80	304
FF80	305
FF80	306
FF80	307
FF80	308
FF80	309
FF80	310
FF80	311
FF80	312
FF80	313
FF80	314
FF80	315C

FF80DY

```

DATA(10) = AMSY(I)
DATA(11) = PSHK(I)
DATA( 2) = TS(I)

C
CALL WRITMS(10, DATA, 2, ITAG10)
C
WRITE(10:ITAG10) DATA
C
ITAG10 = ITAG10 + 1

C
310 CONTINUE
GO TO (120, 140, 170, 400), TK

C
CHECK FOR SECONDARY FLOWS. FIRST SAVE COUNTERS
400 CONTINUE
CALL WRITMS(10, LOC, 25, IG10)
WRITE(10:IG10) LOC

C
IF (NSECF .EQ. 0) GO TO 500

C
SECONDARY FLOW. SET LOCATIONS AND COUNTERS.
ISF = ISF + 1
IG10 = ITAG10
LCNEXT = IG10 + 1
NSECF = NSECF + 1
LOSF(ISF) = IG10
GO TO 100

C
500 CONTINUE
LCNEXT = ITAG10
IF (ISF .EQ. 0) GO TO 1500

C
RESET LOCATIONS FOR SECONDARY FLOWS
IG10 = LORG(IREG) + 4 + N

C
CALL READMS(10, LOC, 25, IG10)

```

FF800Y

FF80 316I
FF80 317
FF80 318
FF80 319
FF80 320
FF80 321C
FF80 322I
FF80 323
FF80 324
FF80 325
FF80 326
FF80 327
FF80 32A
FF80 329
FF80 330
FF80 331
FF80 332
FF80 333
FF80 334C
FF80 335I
FF80 336
FF80 337
FF80 338
FF80 339
FF80 340
FF80 341

```
C READ(10#IG10) LOC
C
C DO 510 I = 1,ISF
  510 LOC(20+I) = (OSF(I)
C CALL WRITMS(10, LOC, 25, IG10)
C WRITE(10#IG10) LOC
C 1500 CONTINUE
C SAVE ORIENTATION DATA
  DAT(1) = YPO
  DAT(2) = YPO
  DAT(3) = ZPO
  DAT(4) = PSIN
  DAT(5) = TMED
  DAT(6) = PHID
  IG10 = LOG(JRFG) + 1
  CALL WRITMS(10, DAT, 6, IG10)
  WRITE(10#IG10) DAT
C
C SET IDTYP(1) FOR NON-UNIFORM FLOW FIELD
  IDTYP(2) = 2
C RETURN
  END
```

FF800Y

FF800Y

FF80	3161
FF80	317
FF80	318
FF80	319
FF80	320
FF80	321C
FF80	3221
FF80	323
FF80	324
FF80	325
FF80	326
FF80	327
FF80	328
FF80	329
FF80	330
FF80	331
FF80	332
FF80	333
FF80	334C
FF80	3351
FF80	336
FF80	337
FF80	338
FF80	339
FF80	340
FF80	341

FF800Y

```

C READ(10#IG10) LOC
C DD 510 I = 1, ISF
  510 LOC(20+I) = LOSF(I)
C CALL WRITMS(10, LOC, 25, IG10)
C WRITE(10#IG10) LOC
C 1500 CONTINUE
C SAVE ORIENTATION DATA
  DAT(1) = XPO
  DAT(2) = YPO
  DAT(3) = ZPO
  DAT(4) = PSIN
  DAT(5) = THEQ
  DAT(6) = PMIO
  IG10 = LORG(JRFG) + 1
  CALL WRITMS(10, DAT, 6, IG10)
  WRITE(10#IG10) DAT
C SET IDTYP(1) FOR NON-UNIFORM FLOW FIELD
C IDTYP(2) = 2
C RETURN
  END

```

SOSE

```

C SURROUTINE SOSE
C SECOND-ORDER SHOCK-EXPANSION METHOD
C BASED ON NACA TR-1328,
C (HIGHLY MODIFIED VERSION OF NASA TN D-5046)
C
COMMON/SOSEIN/ AMF, ALF, PHIT, G, IFLG(5), N, AMP, AAFP,
1 X(100), Y(100)
COMMON/SOSEDT/A/VANS(100),AMDZ(100), AMY(100), AMDZ(100),
1 PANS(100), TD(100), AMS(100), AMSX(100), AMSY(100), AMSZ(100),
2 PSHK(100), TS(100), NSP, NS, JS(10), IS(10), LS(10), IN(10),
3 XANS(100), VANS(100), XS(100), YS(100)
C
COMMON/FSBS/FS,BS
DIMENSION FS(8),RS(8),ANGL(3)
DIMENSION OFL(100), S(100), T(100), C(100)
EQUIVALENCE (DEL(1), C(1))
C
REAL MIDDLE,MNS
DATA RC,DC/1.745329252E-2, 57.29577957/, PMIN/0.3/
DATA MFR,IPRJNT,ISDET,CPSTAG/C,0,0,2,0/
GP1 = G + 1.0
GM1 = G - 1.0
GR = SQRT(GP1/GM1)
ANJMX = 0.97815*(GR - 1.0)*90.0*RC
AMF = AMF
AMP2=AMP**2
ALPHA = DC*ALF
ALFPO = DC*ALFP
PHIT = DC+PHIT
IN(1) = 0
JSS = 0
J = 1
NS = 1
JSJ = 1

```

SOSE 001
SOSE 002
SOSE 003
SOSE 004
SOSE 005
SOSE 006
SOSE 007
SOSE 008
SOSE 009
SOSE 010
SOSE 011
SOSE 012
SOSE 013
SOSE 014
SOSE 015
SOSE 016
SOSE 017
SOSE 018
SOSE 019
SOSE 020
SOSE 021
SOSE 022
SOSE 023
SOSE 024
SOSE 025
SOSE 026
SOSE 027
SOSE 028
SOSE 029
SOSE 030
SOSE 031
SOSE 032
SOSE 033
SOSE 034
SOSE 035

SOSE

S0SE

S0SE 036
 S0SE 037
 S0SE 038
 S0SE 039
 S0SE 040
 S0SE 041
 S0SE 042
 S0SE 043
 S0SE 044
 S0SE 045
 S0SE 046
 S0SE 047
 S0SE 048
 S0SE 049
 S0SE 050
 S0SE 051
 S0SE 052
 S0SE 053
 S0SE 054
 S0SE 055
 S0SE 056
 S0SE 057
 S0SE 058
 S0SE 059
 S0SE 060
 S0SE 061
 S0SE 062
 S0SE 063
 S0SE 064
 S0SE 065
 S0SE 066
 S0SE 067
 S0SE 068
 S0SE 069
 S0SE 070

```

JS (1) = 1
IS(1) = 1

XS(1) = X(1)
YS(1) = Y(1)
NSP = 0
XSMAX = 3.0*X(N)

C
C SET FLAGS FOR FLOW TYPE
IFLG3 = IFLG(3)
IFLG4 = IFLG(4)
IF (IFLG3 .LF. 0) GO TO 7
GO TO (1,2,3,4), IFLG3

C
C FOSF, 2=0 STARTING FLOW
7 IT = 0
IFLG3 = 0
GO TO 5

C
C SOSF, CP 792 CONE FLOW
1 IT = 1
IFLG3 = 1
GO TO 5

C
C S0SE, JONES CONE FLOW
2 IT = 2
IFLG3 = 1
GO TO 5

C
C F0SE, CP 792 CONE STARTING FLOW
3 IT = 1
IFLG3 = 0
GO TO 5

C
C F0SE, JONES CONE STARTING FLOW

```

S0SE

SOSE

SOSE 071
 SOSE 072
 SOSE 073
 SOSE 074
 SOSE 075
 SOSE 076
 SOSE 077
 SOSE 078
 SOSE 079
 SOSE 080
 SOSE 081
 SOSE 082
 SOSE 083
 SOSE 084
 SOSE 085
 SOSE 086
 SOSE 087
 SOSE 088
 SOSE 089
 SOSE 090
 SOSE 091
 SOSE 092
 SOSE 093
 SOSE 094
 SOSE 095
 SOSE 096
 SOSE 097
 SOSE 098
 SOSE 099
 SOSE 100
 SOSE 101
 SOSE 102
 SOSE 103
 SOSE 104
 SOSE 105

```

4 IT = 2
IFLG3 = 0
5 CONTINUE

C CALCULATE LOCAL SLOPES AND RUNNING LENGTHS
IF (IFLG(1) .EQ. 1) GO TO 9

C LINEAR SLOPES AT MIDPT
S(1) = 0.0
DO 6 I = 2,N
  DY = Y(I) - Y(I-1)
  DX = -X(I) + X(I-1)
  DS = SQRT(DX**2 + DY**2)
  S(I) = S(I-1) + DS
6 DEL(T) = DC*ATAN2(DY,DX)
  DEL(1) = DEL(2)
  GO TO 11

C CIRCULAR ARC ROUTINE
9 CALL CADA(N, X, Y, T, S, C)

C CAD RETURNS SLOPE ANGLE (T), LENGTH (S), AND CURVATURE (C)
C AT THE N POINTS == SET DEL AND S AT MIDPOINTS.
C
  DEL(1) = DC*T(1) + 180.0
  DO 10 I = 2,N
    DEL(I) = DC*0.5*(T(I+1) + T(I)) + 180.0
  10 CONTINUE

C
  11 CONTINUE
IF (IFLG(5) .NE. 1) GO TO 12

```

SOSE

SOSE

SOSE 106
 SOSE 107
 SOSE 108
 SOSE 109
 SOSE 110
 SOSE 111
 SOSE 112
 SOSE 113
 SOSE 114
 SOSE 115
 SOSE 116
 SOSE 117
 SOSE 118
 SOSE 119
 SOSE 120
 SOSE 121
 SOSE 122
 SOSE 123
 SOSE 124
 SOSE 125
 SOSE 126
 SOSE 127
 SOSE 128
 SOSE 129
 SOSE 130
 SOSE 131
 SOSE 132
 SOSE 133
 SOSE 134
 SOSE 135
 SOSE 136
 SOSE 137
 SOSE 138
 SOSE 139
 SOSE 140

```

C      WRITE(6,32)
      FOPMAT(1M1)
      IF (IFLG3.EQ. 1) WRITE(6, 33)
      IF (IFLG3.EQ. 0) WRITE(6, 34)
      33 FORMAT(1H ,20H SECOND-ORDER SHOCK-EXPANSION)
      WRITE(6, 51) AMP, ALPHA, PHT, AMP, ALFPO
      51 FORMAT(//5HOM = , F12.6, 5X 8HALFWD = , F12.6,
      15X7HPHT = , F12.6, 5X6HAMP = , F12.6, 5X7HALFP = , F12.6)
      WRITE(6,37)
      37 FORMAT(1H0, 22X1HX, 14X1HY, 13X5HDELTA, 11X1HS/)
      38 WRITE(6,38) (X(I),Y(I),DEL(I),S(I),I=1,N)
      FORMAT(13X, 4F15.4)
C      C STARTING SOLUTION
      12 CONTINUE
      I = 2
      DELC = RC*DEL(I)
      DEL0 = -ALFP
      FSHKO = DELC - DEL0
      IF (IT.NE. 0) GO TO 250
C      C 2-D STARTING FLOW
      ANGL(2) = DEL(I) + ALFPO
      FS(6) = AMP
      FS(2) = 1.0
      FS(3) = 1.0
      IFIRST = 0
      IF (ANGL(2) .LE. 0.0) GO TO 200
C      CALL COMPR(ANGL, MER, IPRINT, CPSTAG, ISDET, IFIRST, CP)
      GO TO 210
      200 ANGL(2) = -ANGL(2)
      CALL EXPAND(ANGL, MER, IPRINT, ISDET, CP)
C

```

SOSE

80SE

```
210 THETA = ANGL(3) = ALFPD
    THETA1 = THETA*RC
    TANS = TAN(THETA1)
    AMANS(I) = BS(6)
    PSHK(J) = BS(2)
    PC = BS(2)
    DPSK = 0.0
    TD(I) = BS(3)
    TS(J) = BS(3)
    COSL = AMP/AME
    ALAM = ARCCOS(COSL)
    WQVS = SIN(ALAM)
    WQVR = WQVS
    VM = AMANS(I)*SQRT(TD(I))/AME
    VSURF = SQRT(VM**2 + WQVR**2)
    AMDX(I) = -VM*COS(DELC)/VSURF
    AMDY(I) = VM*SIN(DELC)/VSURF
    AMDZ(I) = WQVR/VSURF
    AMSX(J) = AMDX(I)
    AMSY(J) = AMDY(I)
    AMSZ(J) = AMDZ(I)
    AMS(J) = AMANS(I)
    T(J) = THETA
    GO TO 300

C CONICAL STARTING FLOW
C 250 CONTINUE
    CALL CONEA(DELC,AME,G,ALF,PHIT,ALFP,AMP,THETA1,AMANS(I),PC,IT)
    SWS = SIN(THETA1 + ALFP)
    MNS = (AMP*SWS)**2
    PSHK(J) = (2.*G*MNS + GM1)/GPI
    DPSK = PC - PSHK(J)
    TANS = TAN(THETA1)
    THETA = DC*THETA1
```

C

```
80SE 141
80SE 142
80SE 143
80SE 144
80SE 145
80SE 146
80SE 147
80SE 148
80SE 149
80SE 150
80SE 151
80SE 152
80SE 153
80SE 154
80SE 155
80SE 156
80SE 157
80SE 158
80SE 159
80SE 160
80SE 161
80SE 162
80SE 163
80SE 164
80SE 165
80SE 166
80SE 167
80SE 168
80SE 169
80SE 170
80SE 171
80SE 172
80SE 173
80SE 174
80SE 175
```

80SE

SDSE

```

C CALCULATE FLOW PROPERTIES BEHIND THE SHOCK
C TEMPERATURE RATIO
  TS(J) = PSHK(J)*(GM1*PSHK(J) + GP1)/(GP1*PSHK(J) + GM1)
C SPEED OF SOUND RATIO
  ASJ = SQRT(TS(J))
C VELOCITY COMPONENTS
  U2 = 1.0 - 2.*(MNS = 1.0)/(GP1*AMP2)
  V2 = 2.*(MNS = 1.0)/(GP1*MNS)*SMS* $\cos(\text{THETA1} + \text{ALFP})$ 
  DEL2 = ATAN2(V2,U2)
  VT2 = SQRT(U2**2 + V2**2)
  UD = VT2* $\cos(\text{DEL2} + \text{ALFP})$ 
  VD = VT2*SIN(DFL2 + ALFP)
  DNS = 1./SQRT(0.5*GP1 + 1./MNS)
  ETA = (1.-0.25*GP1*DNS*(1.+DNS))/(1.+0.25*GP1*(1.-DNS))
  MOVS = (1.-ETA)*ALF*SIN(PHIT)*(1.-ETA* $\cos(\text{PHIT})/\text{TANS}$ )
  MOVB = MOVS
  IF (DELC .LE. 0.0) GO TO 54
  MOVB = MOVS*SMS/SIN(DELC)
54 CONTINUE
  WD = MOVS
  VT = SQRT(UD**2 + VD**2 + WD**2)
  AMS(J) = VT*AME/ASJ
  AMSX(J) = UD/VT
  AMSY(J) = VD/VT
  AMSZ(J) = WD/VT
  T(J) = THETA
C
C CALCULATE SURFACE PROPERTIES
  TD(I) = (1.+0.5*GM1*AME**2)/(1.+0.5*GM1*AMNS(I)**2)
  ASJ = SQRT(TD(I))
  VSURF = AMNS(I)*ASJ/AME
  WD = MOVB
  VM = SQRT(VSURF**2 + WD**2)
  UD = VM* $\cos(\text{DELC})$ 
  VD = VM*SIN(DELC)

```

SDSE

S0SE 176
 S0SE 177
 S0SE 178
 S0SE 179
 S0SE 180
 S0SE 181
 S0SE 182
 S0SE 183
 S0SE 184
 S0SE 185
 S0SE 186
 S0SE 187
 S0SE 188
 S0SE 189
 S0SE 190
 S0SE 191
 S0SE 192
 S0SE 193
 S0SE 194
 S0SE 195
 S0SE 196
 S0SE 197
 S0SE 198
 S0SE 199
 S0SE 200
 S0SE 201
 S0SE 202
 S0SE 203
 S0SE 204
 S0SE 205
 S0SE 206
 S0SE 207
 S0SE 208
 S0SE 209
 S0SE 210

SOSE

SOSE 211
 SOSE 212
 SOSE 213
 SOSE 214
 SOSE 215
 SOSE 216
 SOSE 217
 SOSE 218
 SOSE 219
 SOSE 220
 SOSE 221
 SOSE 222
 SOSE 223
 SOSE 224
 SOSE 225
 SOSE 226
 SOSE 227
 SOSE 228
 SOSE 229
 SOSE 230
 SOSE 231
 SOSE 232
 SOSE 233
 SOSE 234
 SOSE 235
 SOSE 236
 SOSE 237
 SOSE 238
 SOSE 239
 SOSE 240
 SOSE 241
 SOSE 242
 SOSE 243
 SOSE 244
 SOSE 245

SOSE

```

AMDY(I) = UD/VSURF
AMDY(I) = VD/VSURF
AMDZ(I) = WD/VSURF

C 300 CONTINUE
XANS(I) = 0.5*(X(I) + X(I-1))
YANS(I) = 0.5*(Y(I) + Y(I-1))
PANS(I) = PC
CP = (PANS(I)-1.)/(AMF**2*.7)
AMU = ARSIN(1./AMANS(I)) + DEL(I)*RC
XANS(I) = X(I)
YANS(I) = Y(I)
PANS(1) = PANS(2)
AMANS(1) = AMANS(2)
AMDY(1) = AMDY(2)
AMDZ(1) = AMDZ(2)
TD(1) = TD(2)

C SHOCK POSITION
J = 2
XS(J) = XS(J-1)
YS(J) = YS(J-1)
IF (IFLG(4) .EQ. 0) GO TO 55
TANH = TAN(AMU)
XS(J) = (-Y(I)+Y(I-1) + X(I-1)*TANS - X(I)*TANH)/(TANS-TANH)
YS(J) = Y(I-1) - (XS(J)-X(I-1))*TANS
PSHK(J) = PSHK(J-1)
TS(J) = TS(J-1)
AMS(J) = AMS(J-1)
AMSX(J) = AMSX(J-1)
AMSY(J) = AMSY(J-1)
AMSZ(J) = AMSZ(J-1)
NSP = NSP + 1
55 CONTINUE

```

S09E

S09E 246
 S09E 247
 S09E 248
 S09E 249
 S09E 250
 S09E 251
 S09E 252
 S09E 253
 S09E 254
 S09E 255
 S09E 256
 S09E 257
 S09E 258
 S09E 259
 S09E 260
 S09E 261
 S09E 262
 S09E 263
 S09E 264
 S09E 265
 S09E 266
 S09E 267
 S09E 268
 S09E 269
 S09E 270
 S09E 271
 S09E 272
 S09E 273
 S09E 274
 S09E 275
 S09E 276
 S09E 277
 S09E 278
 S09E 279
 S09E 280

S09E

```

AMABAMANS(I)
ANGLE = DC*AMU
DPD9A=0
PAEPC

IF (IFLG(5) .NE. 1) GO TO 60
WRITE(6,36) AMANS(I), THETA, DPSK
36 FORMAT(1H0, 11X,21HSTARTING FLOW = M = , F8.5,
1 5X, 8HTHETA = , F8.3, 5X, 7HDPSK = , F8.5)
WRITE(6,39)
39 FORMAT(1H1, 26HINVISCID SOLUTION, SURFACE//1H ,
1 11X 4HX , 5X 8HP/P(TNF), 5X 1HM, 10X 2MCP, 6X,
2 6X2HP8,11X3HETA,
3 7X, 24HANGLE, CPI M PC)
ETA = 0.0
WRITE(6,40) XANS(I),PANS(I),AMANS(I),CP,PANS(I),FTA,ANGLF, CP,PC
40 FORMAT(7XF8.4, 3XF9.5, 3XF7.4, 3XF7.4, 8XF9.5, 3XF9.5, 3XF8.4,
1 3XF7.4, 3XF9.5)

C
C
C BEGIN MAJOR LOOP
60 CONTINUE
DO 25 I=3,N
D1 = RC*DEL(I-1)
D2 = DEL(I)*RC
ANG = D2 = D1
DFIC = D2
IF (IFLG3 .EQ. 0) GO TO 8
CALL CONFAC(DFIC,AME,G,ALF,PHIT,ALFP,AMP,THETA1,AMC,PC,IT)
8 ANUA=2.4495*ATAN(.40825*SQRT(AMA**2-1.))-ATAN(SQRT(AMA**2-1.))
ANUB = ANUA = ANG
IF (ANUR,GE,ANUA) GO TO 14

C
C IF COMPRESSION ANGLE .LE. 1.0 DEGREE, USE ISENTROPIC EQUATIONS
C AND NO SECONDARY SHOCK FORMATION.

```

80SE

80SE 281
80SE 282
80SE 283
80SE 284
80SE 285
80SE 286
80SE 287
80SE 288
80SE 289
80SE 290
80SE 291
80SE 292
80SE 293
80SE 294
80SE 295
80SE 296
80SE 297
80SE 298
80SE 299
80SE 300
80SE 301
80SE 302
80SE 303
80SE 304
80SE 305
80SE 306
80SE 307
80SE 308
80SE 309
80SE 310
80SE 311
80SE 312
80SE 313
80SE 314
80SE 315

```

IF (ANG .LE. 0.0175) GO TO 14
C
C CALCULATION FOR COMPRESSION AND SHOCK FORMATION
C SFT FLAG FOR SECONDARY SHOCK FORMATION
JSS = 1
IF (IFLG(4) .NE. 0) IFLG4 = 1
C
  ANGL(2) = ANG*DC
  FS(6) = AMA
  FS(2) = PA
  IFIRST = 0
  CALL COMPR(ANGL, MER, IPRINT, CPSTAG, ISNET, IFIRST, CP)
  SKANG = ANGL(3)*RC
  AMR = BS(6)
  PB = BS(2)
  AMUR = STN(1./AMB)
  IF (IFLG3 .EQ. 0) GO TO 15

```

C NOTE. REFERENCE NACA TR-1328 USES ANGLES SIGMA, D1, AND D2
C ALL W.R.T. BODY AXES. HOWEVER THEY APPEAR ONLY AS
C DIFFERENCES (D2-D1), (SIGMA-D1), AND (SIGMA-D2).
C IN THE PRESENT NOTATION THESE ARE DEFINED AS

```

  ANG = D2 - D1
  SKANG = SIGMA - D1
  SKANG = ANG + SIGMA - D2
C
  TDPI = TAN(ANG)
  CSD1 = COS(SKANG)
  SSD1 = SIN(SKANG)
  SSD2 = SIN(SKANG - ANG)
  TANW = TAN(AMUR)
  XX = 2.4*(TDPI*CSD1 - SSD1)*(AMA*SSD1)**2 + SSD1

```

80SE

SOSE

```

C      YY = 1.0 + (1. - 2.*SSD1)**2 + 2.*TD21*SSD1*(CSD1)*(AMA*SSD1)**2
C      F = (1. + 0.2*AMA**2)*SSD1*XX/(0.6*YY)
      RA = 1.4*PA*AMA**2/(2.*(AMA**2-1.))
      RB = 1./PB*AMR**2/(2.*(AMB**2-1.))
      XY = 2.*BB/(Y(I-1))*(SSD1*SIN(D1)/SSD2 + SIN(D2))
      1  + DPDSA*(BB*SSD1/(BA*SSD2) + (PB/PA * F)*CSD1*TANM/SSD2)
C      DPDS = XY/(1. + TANM/TAN(SKANG - ANG))
      15 DELD = D1
      FSHKO = ANG
      THFT1 = SKANG + D1
      GO TO 21
C
C      PRANDTL MEYER EXPANSION (OR COMPRESSION)
      14 CONTINUE
      ANGL(2) = -(ANG)*DC
      FS(6) = AMA
      CALL EXPAND(ANGL, MER, IPRINT, ISDET, CP)
      AMB = BS(6)
      PB = PA*(CP*0.5*G*AMA**2 + 1.0)
C      IF (PB .LT. PMIN) PB = PMIN
      IF (IFLG3 .EQ. 0) GO TO 27
      OMEG12 = AMB/AMA*((1.+5*GM1*AMA**2)/(1.+5*GM1*AMB**2))**(.0,5*
      1  GPI/GM1)
      BA = 0.5*G*PA*AMA**2/(AMA**2 - 1.0)
      BB = 0.5*G*PR*AMB**2/(AMB**2 - 1.0)
      DPDS = BB/Y(I-1)*(OMEG12*SIN(D1) - SIN(D2))
      1  + BB/BA*OMEG12*DPDSA
C      21 IF (IFLG3 .EQ. 0) GO TO 27
      IF (PC .EQ. PR) GO TO 22

```

SOSE 316
 SOSE 317
 SOSE 318
 SOSE 319
 SOSE 320
 SOSE 321
 SOSE 322
 SOSE 323
 SOSE 324
 SOSE 325
 SOSE 326
 SOSE 327
 SOSE 328
 SOSE 329
 SOSE 330
 SOSE 331
 SOSE 332
 SOSE 333
 SOSE 334
 SOSE 335
 SOSE 336
 SOSE 337
 SOSE 338
 SOSE 339
 SOSE 340
 SOSE 341
 SOSE 342
 SOSE 343
 SOSE 344
 SOSE 345
 SOSE 346
 SOSE 347
 SOSE 348
 SOSE 349
 SOSE 350

SOSE

SOSE

```

351 FTA = DPDS*(S(I) - S(I-1))/(PC - PB)
352 IF (FTA .GT. 180.) FTA = 180.
353 IF (FTA) 22,22,23
354
355 C FIRST ORDER SHOCK EXPANSION (2-D)
356 22 IF (JFLG(5) .EQ. 1) WRITE(6, 34)
357 FORMAT (1X25H1ST ORDER SHOCK EXPANSION)
358 27 PANS(I)=PB
359 AMANS(I)=AMB
360 AMASAMP
361 PA=PB
362 IF (JFLG3 .EQ. 0) PC = PB
363 XANS(I)=(X(I)+X(I-1))/2.
364 YANS(I) = 0.5*(Y(I) + Y(I-1))
365 DPDSAE0.
366 GO TO 24
367
368 C SECOND ORDER SHOCK EXPANSION
369 PA=PC -(PC -PB)*EXP(-ETA)
370 AMA=SQRT(2./GM1*((1.+5*GM1*AMR**2)*(PB/PA)**(GM1/G)+ 1.0))
371 XANS(I)=(X(I)+X(I-1))/2.
372 YANS(I) = 0.5*(Y(I) + Y(I-1))
373 ETANS=ETA/2.
374 PANS(I)=PC -(PC -PB)*EXP(-ETANS)
375 AMANS(I)=SQRT(2./GM1*((1.+5*GM1*AMB**2)*(PB/PANS(I))**
376 (GM1/G) + 1.0))
377 1 DPDSAE(PC -PA)/(PC -PB)*DPDS
378 24 CP = 2.*(PANS(I) - 1.0)/(G+AMF**2)
379
380 C CALCULATE SURFACE PROPERTIES
381 TD(I) = (1. +0.5*GM1*AME**2)/(1. + 0.5*GM1*AMANS(I)**2)
382 ASJ = SORT(TD(I))
383 VSURF = AMANS(I)*ASJ/AME
384 WD = WOVE
385 VM = SQRT(VSURF**2 + WD**2)

```

SOSE

SOSE

SOSE 386
 SOSE 387
 SOSE 388
 SOSE 389
 SOSE 390
 SOSE 391
 SOSE 392
 SOSE 393
 SOSE 394
 SOSE 395
 SOSE 396
 SOSE 397
 SOSE 398
 SOSE 399
 SOSE 400
 SOSE 401
 SOSE 402
 SOSE 403
 SOSE 404
 SOSE 405
 SOSE 406
 SOSE 407
 SOSE 408
 SOSE 409
 SOSE 410
 SOSE 411
 SOSE 412
 SOSE 413
 SOSE 414
 SOSE 415
 SOSE 416
 SOSE 417
 SOSE 418
 SOSE 419
 SOSE 420

```

UD = VM*COS(D2)
VD = VM*SIN(D2)
AMDY(I) = UD/VSURF
AMDZ(I) = VD/VSURF
AMOX(I) = WD/VSURF
AMOZ(I) = VD/VSURF
C SHOCK CALCULATION
C
  AMU = ARGIN(1.0, MA) + D2
  ANGLE = DC*AMU
  CPCONE = (PC - 1.0)/(0.7*AMF**2)
  IF (JSS, NE, 1) GO TO 70
C
C SECONDARY SHOCK FORMATION
C SET COUNTER (LSP) AND POSITION FOR LAST SHOCK POINT
  JSS = 0
  LS(NS) = NSP
  LSP = NSP + 1
C SET COUNTER JS FOR START OF NEW SHOCK
C AND SET POSITION EQUAL TO BODY LOCATION
  NS = NS + 1
  J = J + 1
  IS(NS) = I
  JS(NS) = J
  XS(J) = X(I-1)
  YS(J) = Y(I-1)
C SET PROPERTIES EQUAL TO 2-D VALUES (FIRST PRINT OF PREFERRED SHOCK).
  PSHK(J) = PB
  MNS = (GP1*PSHK(J) + GM1)/(2.*G)
  TS(J) = PSHK(J)*(GM1*PSHK(J) + GP1)/(GP1*PSHK(J) + GM1)
  ASJ = SQRT(TS(J))
  UD = COS(D2)
  VD = SIN(D2)
  WD = WCVS
  VT = SQRT(UD**2 + VD**2 + WD**2)
  AMS(J) = VT*AME/ASJ

```

SOSE

808E

808E 421
 808E 422
 808E 423
 808E 424
 808E 425
 808E 426
 808E 427
 808E 428
 808E 429
 808E 430
 808E 431
 808E 432
 808E 433
 808E 434
 808E 435
 808E 436
 808E 437
 808E 438
 808E 439
 808E 440
 808E 441
 808E 442
 808E 443
 808E 444
 808E 445
 808E 446
 808E 447
 808E 448
 808E 449
 808E 450
 808E 451
 808E 452
 808E 453
 808E 454
 808E 455

```

AMX(J) = UD/VT
AMSY(J) = VD/VT
AMSZ(J) = WD/VT
T(J) = THETA1*DC
J = J + 1
NSP = NSP + 3
JSJ = I

C SET UP SECOND POINT OF EMREDFD SHOCK
PSHK(J) = PSHK(J=1)
AMS(J) = AMS(J=1)
AMX(J) = AMX(J=1)
AMSY(J) = AMSY(J=1)
AMSZ(J) = AMSZ(J=1)
THETA = THETA1
DPSK = 0.0
SWS = SIN(THETA1 + ALFP)
GD TO 72

C
C 70 IF (IFLG4 .EQ. 0) GO TO 80
J = J + 1
FSHK = 1.0
FSHKI = D2 - DELD
IF (FSHK0 .GT. 0.0) FSHK = FSHKI/FSHK0
PSHK(J) = PANS(I) - DPSK*FSHK
IF (PSHK(J) .LT. 1.0) PSHK(J) = 1.0
C CALCULATE FLOW PROPERTIES BEHIND THE SHOCK
MNS = (GPI*PSHK(J) + GM1)/(2.*G)
THETA = ARSIN(SQRT(MNS/AMP2)) = ALFP
TS(J) = PSHK(J) + (GM1*PSHK(J) + GM1)/(GPI*PSHK(J) + GM1)
ASJ = SQRT(TS(J))
SWS = SIN(THETA)
U2 = 1.0 - 2.*(MNS - 1.0)/(GPI*AMP2)
V2 = 2.*(MNS + 1.0)/(GPI*MNS)*SWS*COS(THETA)
  
```

808E

SOSE

SOSE 456
SOSE 457
SOSE 458
SOSE 459
SOSE 460
SOSE 461
SOSE 462
SOSE 463
SOSE 464
SOSE 465
SOSE 466
SOSE 467
SOSE 468
SOSE 469
SOSE 470
SOSE 471
SOSE 472
SOSE 473
SOSE 474
SOSE 475
SOSE 476
SOSE 477
SOSE 478
SOSE 479
SOSE 480
SOSE 481
SOSE 482
SOSE 483
SOSE 484
SOSE 485
SOSE 486
SOSE 487
SOSE 488
SOSE 489
SOSE 490

SOSE

```

DEL2 = ATAN2(V2,U2)
VT2 = SQRT(UP**2 + V2**2)
UD = VT2*COS(DEL2 + ALFP)
VD = VT2*SIN(DEL2 + ALFP)
WD = WOV8
VT = SQRT(UD**2 + VD**2 + WD**2)
AMS(J) = VT*AME/ASJ
AMSX(J) = UD/VT
AMSY(J) = VD/VT
AMSZ(J) = WD/VT
T(J-1) = THETA*OC

C 72 CONTINUE
IF (THETA .LT. AMU) GO TO 73
J = J + 1
IFLG4 = 0
GO TO 80

73 CONTINUE
TANM = TAN(AMU)
TANS = TAN(THETA)
XS(J) = (Y(I) + YS(J-1) + XS(J-1))*TANS + X(I)*TANM)/(TANS-TANM)
YS(J) = YS(J-1) + (XS(J) - XS(J-1))*TANS
NSP = NSP + 1
IF (XS(J) .GT. XSMAX) GO TO 74
C HAVE REACHED LIMIT FOR SHOCK, TURN OFF FLAG 4
IFLG4 = 0
74 CONTINUE
IF (JSS .NE. 2) GO TO 60

C 75 CONTINUE
C TWO SHOCKS, CHECK FOR INTERSECTION
L = LSP
76 CONTINUE
IF (XS(J) .LT. XS(L)) GO TO 77
L = L + 1
GO TO 76

```

808E

```

C      77 IF (YS(J) *LT, YS(L)) GO TO 80
C
C      C INTERSECTION EXISTS. DETERMINE SECOND POINT ON
C      C FIRST SHOCK TO BE USED IN CALCULATION.
      L1 = L
      L2 = L + 1
      IF (XS(J-1) .LT. XS(L)) GO TO 7A
      L1 = L - 1
      L2 = L
      7A CONTINUE
C
      TANH = TANS
      TANS = (YS(L2) - YS(L1))/(XS(L2) - XS(L1))
      IF (TANS .GE. TANH) GO TO 75
C
C      XSI = (-YS(J)+YS(J-1)+XS(L1)*TANS+XS(J)*TANH)/(TANS+TANH)
C
C      C SET LAST SHOCK POINT COUNTER AND RESET COORDS.
      LS(NS-1) = L2
      XS(L2) = XSI
      YS(L2) = YS(L1) - (XSI - XS(L1))*TANS
      IN(NS-1) = J-1
C
      75 JSS = 0
      80 CONTINUE
C
      IF (IFLG(5) .EQ. 1)
        AWRITE(6,40) XANS(I), PANS(I), AWANS(I), CF, PB, ETA, ANGLE, CPCONE, PC
      25 CONTINUE
C
      LS(NS) = NSP
C
C      WRITE OUT SHOCK DATA

```

808E

80SE

```
C
DO 110 I = 1, N8
IF (IFLG(5) .NE. 1) GO TO 89
WRITE(6,99) I
90 FORMAT(1M1,30HINVISCID SOLUTION, SHOCK WAVE .I3//1M ,
1 12X2HXS, 12X2HYS, 11X5HP2/P1, 10X2HM2, 10X5HTHETA/)
89 CONTINUE
J1 = J3(I)
J2 = L8(I)
C
C CHECK IF AT LEAST THREE SHOCK POINTS
IF ((J2-J1) = 1) 91,92,93
C
C ONLY ONE SHOCK POINT, ADD ONE AT XSMAX
91 J2 = J1 + 1
XS(J2) = XSMAX
YS(J2) = Y8(J1) + (XS(J1)-XS(J2))*TAN(PC*T(J1))
T(J2) = T(J1)
AMS(J2) = AMS(J1)
AMX(J2) = AMX(J1)
AMSY(J2) = AMSY(J1)
AMSZ(J2) = AMSZ(J1)
PSHK(J2) = PSHK(J1)
TS(J2) = TS(J1)
C
C ONLY TWO SHOCK POINTS, ADD ONE AT MIDPOINT
92 J3 = J1 + 2
XS(J3) = XS (J2)
YS(J3) = YS(J2)
T(J3) = T(J2)
AMS(J3) = AMS(J2)
AMX(J3) = AMX(J2)
AMSY(J3) = AMSY(J2)
AMSZ(J3) = AMSZ(J2)
PSHK(J3) = PSHK(J2)
```

80SE 526
80SE 527
80SE 528
80SE 529
80SE 530
80SE 531
80SE 532
80SE 533
80SE 534
80SE 535
80SE 536
80SE 537
80SE 538
80SE 539
80SE 540
80SE 541
80SE 542
80SE 543
80SE 544
80SE 545
80SE 546
80SE 547
80SE 548
80SE 549
80SE 550
80SE 551
80SE 552
80SE 553
80SE 554
80SE 555
80SE 556
80SE 557
80SE 558
80SE 559
80SE 560

80SE

8092

809E 561
809E 562
809E 563
809E 564
809E 565
809E 566
809E 567
809E 568
809E 569
809E 570
809E 571
809E 572
809E 573
809E 574
809E 575
809E 576
809E 577
809E 578
809E 579
809E 580
809E 581
809E 582
809E 583
809E 584

```

C      TS(J3) = TS(J2)
      XS(J2) = 0.5*(XS(J3) + XS(J1))
      YS(J2) = 0.5*(YS(J3) + YS(J1))
      T(J2)  = 0.5*(T(J3) + T(J1))
      AMS(J2) = 0.5*(AMS(J3) + AMS(J1))
      AMSX(J2) = 0.5*(AMSX(J3) + AMSX(J1))
      AMSY(J2) = 0.5*(AMSY(J3) + AMSY(J1))
      AMSZ(J2) = 0.5*(AMSZ(J3) + AMSZ(J1))
      PSHK(J2) = 0.5*(PSHK(J3) + PSHK(J1))
      TS(J2)  = 0.5*(TS(J3) + TS(J1))
C      LS(I) = J3
C
C      93 J2 = L6(I)
      IF (IFLG(5) .NE. 1) GO TO 110
      DO 100 J = J1,J2
      WRITE(6,95) XS(J), YS(J), PSHK(J), AMS(J), T(J)
      95 FORMAT(1H, 3X, 3XF10.4, 4(5XF9.4))
      100 CONTINUE
      110 CONTINUE
      RETURN
      END

```

809E

SHIELD

SHIE 001C
 SHIE 002C
 SHIE 003I
 SHIE 004
 SHIE 005
 SHIE 006
 SHIE 007
 SHIE 008
 SHIE 009
 SHIE 010
 SHIE 011
 SHIE 012
 SHIE 013
 SHIE 014
 SHIE 015
 SHIE 016
 SHIE 017
 SHIE 018
 SHIE 019
 SHIE 020
 SHIE 021
 SHIE 022
 SHIE 023
 SHIE 024
 SHIE 025
 SHIE 026
 SHIE 027
 SHIE 028
 SHIE 029
 SHIE 030
 SHIE 031
 SHIE 032
 SHIE 033
 SHIE 034
 SHIE 035

```

OVERLAY (MARK4,2,2)
PROGRAM SHIELD
SUBROUTINE SHIELD

C THIS ROUTINE IS USED TO ACCOUNT FOR SHIELDING OF ONE ELEMENT BY
C ANOTHER.
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOT, TAPE1, TAPEJ, TAPE6, TAPEFC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /TAGS/ITAG4, ITAG9, ITAG10
COMMON /ABDATA/NAB, ALPHA(20), BETA(20), POL(20), CDFLTA(20), QI(20),
1 RI(20), PI(20)
COMMON /INTERF/INT, ISHE(20), NSHE(100), I9, ISHEF, INF(20,22),
1 DINF(20,6), LTOTAB(20)
DIMENSION TITLE(15), IN(500), IM(500), NX(500), NY(500), NZ(500),
1 APEA(500), X1(500), X2(500), X3(500), X4(500), Y1(500), Y2(500),
2 Y3(500), Y4(500), Z1(500), Z2(500), Z3(500), Z4(500), ELEM(25), XS(5),
3 Y(5), Z(5), Y2(5), Z2(5), S(4,2), L8(4,5), INT1(8), INT2(8),
4 S2(2), LS2(2), XP(8), YP(8), ZP(8), X(5), Y2(4), IPF(9), XI(5), YI(5),
5 ZI(5), YPIE(9), ZPIE(9), XSID(2), YSID(2), ZSID(2), YSIP(2), ZSIP(2),
6 XN(4,3), YO(4,3), ZO(4,3), XPA(4), YPA(4), ZPA(4), XII(4), ETA(4),
7 XCENT(500), YCENT(500), ZCENT(500), XC(3), YC(3), ZC(3), AR(3), F(25),
8 IV1(4), EM(25), EP(25)

C
C
INTEGER CASE, PAGE, ERROR, SYMFCT
INTEGER TAPEIN, TAPEOT, TAPE1, TAPEJ, TAPE6, TAPEFC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
REAL NX, NY, NZ, NX0, NX02, LS, LS2, NX2, NY2, NZ2, NDN2
DATA I8IZE/500/

REWIND TAPEA
REWIND TAPEB

```

SHIELD

SHIELD

SHIE	036
SHIE	037
SHIE	038
SHIE	039
SHIE	040
SHIE	041
SHIE	042C
SHIE	043I
SHIE	044
SHIE	045
SHIE	046
SHIE	047
SHIE	048
SHIE	049
SHIE	050
SHIE	051
SHIE	052
SHIE	053
SHIE	054
SHIE	055
SHIE	056C
SHIE	057I
SHIE	058
SHIE	059
SHIE	060
SHIE	061
SHIE	062
SHIE	063
SHIE	064
SHIE	065
SHIE	066
SHIE	067
SHIE	068
SHIE	069
SHIE	070C

```

19 = 0
DO 10 I=1,100
10 NSHF(I) = 0
20 KMAT (I2,I1,I2,I1,15A4)
READ (TAPEIN,20) NPANL,IPRINT,INAB,ITIE
IG4 = 2
CALL READMS (4,EM,25,IG4)
PEAD (4#IG4) EM
NEXT = EM(1)
NP = EM(2)
NPMAX = EM(3)
NRFM = EM(4)
NPAN = 0
30 NPAN = NPAN + 1
40 READ (TAPEIN,50) IPAN,ISHE
50 FORMAT (I2,20I2)

C
C
C READ PANEL TO BE ANALYZED FOR SHIELDING
IG4 = IPAN*5
CALL READMS (4,EP,25,IG4)
READ (4#IG4) EP
ISTAT3 = EP(1)
COM = EP(2)
ISTART = EP(3)
L = EP(4)
IORN = EP(5)
SYMFACT = EP(6)
ISMY = SYMFACT
ISTART = ISTART - NREM
IF (L.GT.ISIZE) GO TO 1030

C
DO 60 I=1,L
IG4 = ISTART + NREM*I
CALL READMS (4,ELEM,25,IG4)

```

SHIELD

SHIELD

SHIE	0711
SHIE	072
SHIE	073
SHIE	074
SHIE	075
SHIE	076
SHIE	077
SHIE	078
SHIE	079
SHIE	080
SHIE	081
SHIE	082
SHIE	083
SHIE	084
SHIE	085
SHIE	086
SHIE	087
SHIE	088
SHIE	089
SHIE	090
SHIE	091
SHIE	092
SHIE	093
SHIE	094
SHIE	095
SHIE	096
SHIE	097
SHIE	098
SHIE	099
SHIE	100
SHIE	101
SHIE	102
SHIE	103C
SHIE	104I
SHIE	105

SHIELD

```

C      READ (4#IG4) ELEM
      IN(I) = ELEM(2) + 0.0001
      IM(I) = ELEM(3) + 0.0001
      NX(I) = ELEM(4)
      NY(I) = ELEM(5)
      NZ(I) = ELEM(6)
      XCEN(I) = ELEM(7)
      YCEN(I) = ELEM(8)
      ZCEN(I) = ELEM(9)
      AREA(I) = ELEM(10)
      X1(I) = ELEM(11)
      X2(I) = ELEM(12)
      X3(I) = ELEM(13)
      X4(I) = ELEM(14)
      Y1(I) = ELEM(15)
      Y2(I) = ELEM(16)
      Y3(I) = ELEM(17)
      Y4(I) = ELEM(18)
      Z1(I) = ELEM(19)
      Z2(I) = ELEM(20)
      Z3(I) = ELEM(21)
      Z4(I) = ELEM(22)
      60 CONTINUE

C      LTOT = L
C
C      READ ALL SHIELDING ELEMENT COMPONENTS AND STORE ON TAPE
      K = 0
      DO 80 I=1,20
      IF (ISHE(I) .EQ. 0) GO TO 90
      IG4 = ISHE(I)+5
      CALL READMS (4,EP,25,IG4)
      READ (4#IG4) EP
      ISTAT3 = EP(I)

```

SHIELD

SHIE 106
 SHIE 107
 SHIE 108
 SHIE 109
 SHIE 110
 SHIE 111
 SHIE 112
 SHIE 113
 SHIE 114
 SHIE 115
 SHIE 116C
 SHIE 117I
 SHIE 118
 SHIE 119
 SHIE 120
 SHIE 121
 SHIE 122
 SHIE 123
 SHIE 124
 SHIE 125
 SHIE 126
 SHIE 127
 SHIE 128
 SHIE 129
 SHIE 130
 SHIE 131
 SHIE 132
 SHIE 133
 SHIE 134
 SHIE 135
 SHIE 136
 SHIE 137
 SHIE 138
 SHIE 139
 SHIE 140

```

COM = EP(2)
ISTART = EP(3)
L = EP(4)
IORN = EP(5)
SYMFCT = EP(6)
ISTART = ISTART + NREM

C
DO 70 IJ=1,L
  K = K + 1
  IGO = ISTART + NREM*IJ
  CALL READMS (4,ELEM,25,IG4)
  READ (QMIG4) ELEM
  WRITE (TAPE8) ELEM
70 CONTINUE
80 CONTINUE

C
90 LYOTS = K

C
C
C
C
START ANGLE OF ATTACK = BETA CYCLE
DO 1020 I=1,NAP
  IF (YPRINT .EQ. 0) GO TO 110
  CALL HEADER
  WRITE (TAPFOT,100) IPAN,ISHE,ALPHA(I),BETA(I)
100 FORMAT (1H0,5IH***SHIELDING PROGRAM OUTPUT NEGATIVE AREA ELEMENTS
  1 /1H ,5X,6MPANEL=,12,3X,17MSHIELDING PANELS=,20I4,/1H ,
  2 5X,6HALPHAE,F6.2,7H BETA=,F6.2,/1H0,8X,12H ICT IN 1M,9X,2HMX,
  3 12X,2HNY,12X,2HNZ,11X,5HXCENT,9X,5HZCENT,10X,4HAREA,
  4 /1H ,29X,2HX1,12X,2HX2,12X,2HX3,12X,2HX4,/1H ,29X,2HY1,12X,2HY2,
  5 12X,2HY3,12X,2HY4,/1H ,29X,2HZ1,12X,2HZ2,12X,2HZ3,12X,2HZ4)
110 CONTINUE
PSI = BETA(I) / 57.2958
THETA = ALPHA(I) / 57.2958
PHI = 0.0
ICT = 0
  
```

SHIELD

SHIELD

SHIE	141
SHIE	142
SHIE	143
SHIE	144
SHIE	145
SHIE	146
SHIE	147
SHIE	148
SHIE	149
SHIE	150
SHIE	151
SHIE	152
SHIE	153
SHIE	154
SHIE	155
SHIE	156
SHIE	157
SHIE	158
SHIE	159
SHIE	160
SHIE	161
SHIE	162
SHIE	163
SHIE	164
SHIE	165
SHIE	166
SHIE	167
SHIE	168
SHIE	169
SHIE	170
SHIE	171
SHIE	172
SHIE	173
SHIE	174
SHIE	175

SHIELD

```

C
C
C   SET UP CONSTANTS FOR ROTATION EQUATIONS
      SINTH = SIN(THETA)
      COSTH = COS(THETA)
      SINPSI = SIN(PHI)
      COSPSI = COS(PHI)
      SINPHI = SIN(PHI)
      COSPHI = COS(PHI)
      A1 = COSTH * SINPSI
      A2 = (COSPSI * COSPHI) + SINTH * (SINPSI * SINPHI)
      A3 = -(COSPSI * SINPHI) + SINTH * (SINPSI * COSPHI)
      A4 = -SINTH
      A5 = COSTH * SINPHI
      A6 = COSTH * COSPHI
      A7 = COSTH * COSPSI
      A8 = -(SINPSI * COSPHI) + SINTH * (COSPSI * SINPHI)
      A9 = (SINPSI * SINPHI) + SINTH * (COSPSI * COSPHI)
C
C
C   SELF-SHIELDING WITHIN COMPONENT WILL BE CHECKED
      LMAX = LTOT + LTOTS
C
      DO 1010 J=1,LTOT
      CRFF = 1.0
      IREFI = 1
      IF (AREA(J) .LT. 0.0001) GO TO 1010
      XI(1) = X1(J)
      XI(2) = X2(J)
      XI(3) = X3(J)
      XI(4) = X4(J)
      YI(1) = Y1(J)
      YI(2) = Y2(J)
      YI(3) = Y3(J)
      YI(4) = Y4(J)

```

SHIELD

SHIE 176
 SHIE 177
 SHIE 178
 SHIE 179
 SHIE 180
 SHIE 181
 SHIE 182
 SHIE 183
 SHIE 184
 SHIE 185
 SHIE 186
 SHIE 187
 SHIE 188
 SHIE 189
 SHIE 190
 SHIE 191
 SHIE 192
 SHIE 193
 SHIE 194
 SHIE 195
 SHIE 196
 SHIE 197
 SHIE 198
 SHIE 199
 SHIE 200
 SHIE 201
 SHIE 202
 SHIE 203
 SHIE 204
 SHIE 205
 SHIE 206
 SHIE 207
 SHIE 208
 SHIE 209
 SHIE 210

SHIELD

```

    ZI(1) = Z1(J)
    ZI(2) = Z2(J)
    ZI(3) = Z3(J)
    ZI(4) = Z4(J)
    GO TO 130
120  XI(2) = X4(J)
    XI(4) = X2(J)
    YI(1) = -Y1(J)
    YI(2) = -Y4(J)
    YI(3) = -Y3(J)
    YI(4) = -Y2(J)
    ZI(2) = Z4(J)
    ZI(4) = Z2(J)
130  LSH = 5
    REWIND TAPES
C  APPLY ROTATION MATRIX TO OUTWARD NORMAL
    NXO = NX(J)*A7 + NY(J)*A8*CREF + NZ(J)*A9
    IF (NXO .LE. 0.0001) GO TO 1000
C  APPLY ROTATION MATRIX TO ELEMENT
    X(1) = XI(1)*A7 + YI(1)*A8 + ZI(1)*A9
    X(2) = XI(2)*A7 + YI(2)*A8 + ZI(2)*A9
    X(3) = XI(3)*A7 + YI(3)*A8 + ZI(3)*A9
    X(4) = XI(4)*A7 + YI(4)*A8 + ZI(4)*A9
    Y(1) = XI(1)*A1 + YI(1)*A2 + ZI(1)*A3
    Y(2) = XI(2)*A1 + YI(2)*A2 + ZI(2)*A3
    Y(3) = XI(3)*A1 + YI(3)*A2 + ZI(3)*A3
    Y(4) = XI(4)*A1 + YI(4)*A2 + ZI(4)*A3
    Z(1) = XI(1)*A4 + YI(1)*A5 + ZI(1)*A6
    Z(2) = XI(2)*A4 + YI(2)*A5 + ZI(2)*A6
    Z(3) = XI(3)*A4 + YI(3)*A5 + ZI(3)*A6
    Z(4) = XI(4)*A4 + YI(4)*A5 + ZI(4)*A6
    YMIN = AMINI (Y(1),Y(2),Y(3),Y(4))
    YMAX = AMAXI (Y(1),Y(2),Y(3),Y(4))
    ZMIN = AMINI (Z(1),Z(2),Z(3),Z(4))
    ZMAX = AMAXI (Z(1),Z(2),Z(3),Z(4))
  
```

SHIELD

```
C
C
C   DO LOOP TO FIND SHIELDING ELEMENTS
C   DO 990 K=1,LMAX
C   IS THIS THE COMPONENT OR IS IT THE SHIELDING ELEMENTS
      IREF2 = 1
      CREF2 = 1.0
C   AVOID ELEMENT SELF-SHIELDING
      140 IF (K.FG.J .AND. IREF1.EQ.IREF2) GO TO 940
      IF (K .GT. LTOT) GO TO 160
      IF (AREA(K) .LT. 0.0001) GO TO 990
C   APPLY ROTATION MATRIX TO NORMAL
      NX02 = NX(K)*A7 + NY(K)*A8+CREF2 + NZ(K)*A9
C   CHECK IF REAR FACING
      IF (NX02 .LE. 0.0001) GO TO 980
C
      ELEM(1) = K
      ELEM(2) = IN(K)
      ELEM(3) = IM(K)
      ELEM(4) = NX(K)
      ELEM(5) = NY(K)
      ELEM(6) = NZ(K)
      ELEM(7) = XCFNT(K)
      ELEM(8) = YCFNT(K)
      ELEM(9) = ZCFNT(K)
      ELEM(10) = ARFA(K)
      ELEM(11) = X1(K)
      ELEM(12) = X2(K)
      ELEM(13) = X3(K)
      ELEM(14) = X4(K)
      ELEM(15) = Y1(K)
      ELEM(16) = Y2(K)
      ELEM(17) = Y3(K)
      ELEM(18) = Y4(K)
      ELEM(19) = Z1(K)
```

```
SHIE 211
SHIE 212
SHIE 213
SHIE 214
SHIE 215
SHIE 216
SHIE 217
SHIE 218
SHIE 219
SHIE 220
SHIE 221
SHIE 222
SHIE 223
SHIE 224
SHIE 225
SHIE 226
SHIE 227
SHIE 228
SHIE 229
SHIE 230
SHIE 231
SHIE 232
SHIE 233
SHIE 234
SHIE 235
SHIE 236
SHIE 237
SHIE 238
SHIE 239
SHIE 240
SHIE 241
SHIE 242
SHIE 243
SHIE 244
SHIE 245
```

SHIELD

SHIELD

SHIE 246
 SHIE 247
 SHIE 248
 SHIE 249
 SHIE 250
 SHIE 251
 SHIE 252
 SHIE 253
 SHIE 254
 SHIE 255
 SHIE 256
 SHIE 257
 SHIE 258
 SHIE 259
 SHIE 260
 SHIE 261
 SHIE 262
 SHIE 263
 SHIE 264
 SHIE 265
 SHIE 266
 SHIE 267
 SHIE 268
 SHIE 269
 SHIE 270
 SHIE 271
 SHIE 272
 SHIE 273
 SHIE 274
 SHIE 275
 SHIE 276
 SHIE 277
 SHIE 278
 SHIE 279
 SHIE 280

```

FLEM(20)=Z2(K)
ELEM(21)=Z3(K)
ELEM(22)=Z4(K)
IF (IREF2 .EQ. 1) GO TO 180
150 FLEM(5) = WY(K)
    FLEM(8) = WYCENT(K)
    FLEM(12) = X4(K)
    FLEM(14) = X2(K)
    FLEM(15) = WY1(K)
    FLEM(16) = WY4(K)
    FLEM(17) = WY3(K)
    FLEM(18) = WY2(K)
    FLEM(20) = Z4(K)
    FLEM(22) = Z2(K)
GO TO 180

C SHIELDING COMPONENTS
160 IF (IREF2 .EQ. 1) READ (TAPER) ELEM
IF (ELEM(10) .LT. 0.0001) GO TO 990
IF (IREF2 .EQ. 1) GO TO 170
ELEM( 5) =ELEM( 5)
ELEM( 8) =ELEM( 8)
ELEM(15) =ELEM(15)
ELEM(16) =ELEM(16)
ELEM(17) =ELEM(17)
ELEM(18) =ELEM(18)
ELT = ELEM(12)
ELEM(12) = ELEM(14)
ELEM(14) = ELT
ELT = FLEM(16)
ELEM(16) = ELEM(18)
ELEM(18) = ELT
ELT = ELEM(20)
ELEM(20) = ELEM(22)
ELEM(22) = ELT
  
```

SHIELD

```

C APPLY ROTATION MATRIX TO NORMAL
170 NXO2 = ELEM(4)*A7 + ELEM(5)*A8 + FLEM(6)*A9
C CHECK IF REAR FACING
IF (NXO2 .LE. 0.0001) GO TO 980
C
180 CONTINUE
YS(1) = ELEM(11)*A1 + FLEM(15)*A2 + FLEM(19)*A3
YS(2) = ELEM(12)*A1 + ELEM(16)*A2 + FLEM(20)*A3
YS(3) = ELEM(13)*A1 + ELEM(17)*A2 + FLEM(21)*A3
YS(4) = ELEM(14)*A1 + ELEM(18)*A2 + FLEM(22)*A3
ZS(1) = ELEM(11)*A4 + ELEM(15)*A5 + FLEM(19)*A6
ZS(2) = ELEM(12)*A4 + ELEM(16)*A5 + FLEM(20)*A6
ZS(3) = ELEM(13)*A4 + ELEM(17)*A5 + FLEM(21)*A6
ZS(4) = ELEM(14)*A4 + ELEM(18)*A5 + FLEM(22)*A6
YMIN2 = AMIN1 (YS(1),YS(2),YS(3),YS(4))
YMAX2 = AMAX1 (YS(1),YS(2),YS(3),YS(4))
ZMIN2 = AMIN1 (ZS(1),ZS(2),ZS(3),ZS(4))
ZMAX2 = AMAX1 (ZS(1),ZS(2),ZS(3),ZS(4))
C
C CHECK FOR POSSIBLE SHIELDING
IF ((YMIN2-YMAX) .GE. 0.0) GO TO 980
IF ((YMIN-YMAX2) .GE. 0.0) GO TO 980
IF ((ZMIN-ZMAX) .GE. 0.0) GO TO 980
IF ((ZMIN-ZMAX2) .GE. 0.0) GO TO 980
C CHECK IF ELEMENT 2 IS REALLY IN FRONT OF ELEMENT 1
XMTN = AMIN1(X(1),X(2),X(3),X(4))
XS(1) = ELEM(11)*A7 + ELEM(15)*A8 + FLEM(19)*A9
XS(2) = ELEM(12)*A7 + ELEM(16)*A8 + ELEM(20)*A9
XS(3) = ELEM(13)*A7 + FLEM(17)*A8 + FLEM(21)*A9
XS(4) = ELEM(14)*A7 + ELEM(18)*A8 + FLEM(22)*A9
XMAX2 = AMAX1(XS(1),XS(2),XS(3),XS(4))
IF (XMIN .GE. XMAX2) GO TO 980
C
C POSSIBLE OVERLAP. CALCULATE SIDE VECTORS OF SHIELDED ELEMENT

```

SHIE 316
 SHIE 317
 SHIE 318
 SHIE 319
 SHIE 320
 SHIE 321
 SHIE 322
 SHIE 323
 SHIE 324
 SHIE 325
 SHIE 326
 SHIE 327
 SHIE 328
 SHIE 329
 SHIE 330
 SHIE 331
 SHIE 332
 SHIE 333
 SHIE 334
 SHIE 335
 SHIE 336
 SHIE 337
 SHIE 338
 SHIE 339
 SHIE 340
 SHIE 341
 SHIE 342
 SHIE 343
 SHIE 344
 SHIE 345
 SHIE 346
 SHIE 347
 SHIE 348
 SHIE 349
 SHIE 350

```

S(1,1) = Y(2) = Y(1)
S(1,2) = Z(2) = Z(1)
S(2,1) = Y(3) = Y(2)
S(2,2) = Z(3) = Z(2)
S(3,1) = Y(4) = Y(3)
S(3,2) = Z(4) = Z(3)
S(4,1) = Y(1) = Y(4)
S(4,2) = Z(1) = Z(4)

C CALCULATE L*ΔS
IC = 0
DO 200 I1=1,4
CHECK FOR ZERO LENGTH SIDE OF FLAMENT 1
IF (S(I1,1).EQ.0.0 .AND. S(I1,2).EQ.0.0) IC = I1
DO 190 I2=1,4
LS(I1,I2) = (ZS(I2)-Z(I1))*S(I1,1) - (YS(I2)-Y(I1))*S(I1,2)
190 CONTINUE
LS(I1,5) = LS(I1,1)
200 CONTINUE

C
DO 210 IE=1,4
INT1(IE) = 0
210 INT2(IE) = 0

C
JPOS = 0
IE = 0
DO 230 I1=1,4
IF (I1 .EQ. IC) GO TO 230
DO 220 I2=1,4
IF (LS(I1,I2).GE.0.0 .AND. LS(I1,I2+1).GE. 0.0) GO TO 220
IF (LS(I1,I2).LE.0.0 .AND. LS(I1,I2+1).LE.0.0) GO TO 220
X(5) = X(I)
Y(5) = Y(I)
Z(5) = Z(I)
YS(5) = YS(I)
  
```

SHIELD

SHIE 351
SHIE 352
SHIE 353
SHIE 354
SHIE 355
SHIE 356
SHIE 357
SHIE 358
SHIE 359
SHIE 360
SHIE 361
SHIE 362
SHIE 363
SHIE 364
SHIE 365
SHIE 366
SHIE 367
SHIE 368
SHIE 369
SHIE 370
SHIE 371
SHIE 372
SHIE 373
SHIE 374
SHIE 375
SHIE 376
SHIE 377
SHIE 378
SHIE 379
SHIE 380
SHIE 381
SHIE 382
SHIE 383
SHIE 384
SHIE 385

```
ZS(5) = ZS(1)
XI(5) = XI(1)
YI(5) = YI(1)
ZI(5) = ZI(1)

C INTERSECTION IS POSSIBLE, DO REVERSE-L CHECK TO VERIFY INTERSECTION.
C (JPOS IS THE #POSSIBLE INTERSECTION# COUNTER)
  JPOS = JPOS + 1
  S2(1) = YS(I2+1) - YS(I2)
  S2(2) = ZS(I2+1) - ZS(I2)
  IF (S2(1).EQ.0.0 .AND. S2(2).EQ.0.0) GO TO 220
  LS2(1) = (Z(I1)-ZS(I2))*S2(1)-(Y(I1)-YS(I2))*S2(2)
  LS2(2) = (Z(I1+1)-ZS(I2))*S2(1)-(Y(I1+1)-YS(I2))*S2(2)
  IF (LS2(1).GE.0.0 .AND. LS2(2).GE.0.0) GO TO 220
  IF (LS2(1).LE.0.0 .AND. LS2(2).LE.0.0) GO TO 220

C A DEFINITE INTERSECTION HAS BEEN CONFIRMED. SET INTERSECTION ID
  IE = IE + 1
  INT1(IE) = I1
  INT2(IE) = I2
  IF (IE.NE.2) GO TO 220
  IF (INT1(1).NE. INT1(2)) GO TO 220
  IF (INT2(1).GT. INT2(1)) GO TO 220
  INT2S = INT2(1)
  INT2(1) = INT2(2)
  INT2(2) = INT2S

C 220 CONTINUE
C 230 CONTINUE

C CHECK IF THE FIRST ELEMENT IS COMPLETELY CONTAINED BY THE SECOND
C (JPOS = 8 BUT INT#S = 0)
  IBP = 0
  IF (JPOS.NE.8) GO TO 280
```

SHIELD

SHIELD

```
      IF (INT1(I) .NE. 0) GO TO 280
      I4 = 0
      DO 260 I1=1,4
      IF (I1 .EQ. IC) GO TO 260
      DO 250 I2=1,4
      S2(I) = YS(I2+1) * YS(I2)
      S2(P) = ZS(I2+1) * ZS(I2)
      IF (S2(I) .NE. 0.0 .AND. S2(P) .NE. 0.0) GO TO 240
      GO TO 250
240 LSP(I) = (Z(I1)+ZS(I2))*S2(I) - (Y(I1)+YS(I2))*S2(2)
      IF (LS2(I) .GT. 0.0) GO TO 260
250 CONTINUE
      I4 = I4 + 1
260 CONTINUE
      IF (IC .EQ. 0 .AND. I4 .NE. 4) GO TO 980
      IF (IC .NE. 0 .AND. I4 .NE. 3) GO TO 980
C
C THE FIRST ELEMENT IS CONTAINED BY THE SECOND, SET UP AND SAVE
C SHIELDED ELEMENT CHARACTERISTICS (THEY ARE THE SAME AS THE 1ST)
      NE = 1
      DO 270 I1=1,4
      IPF(I1) = I1*100
      XC(I1,1) = XI(I1)
      XP(I1) = XI(I1)
      YP(I1,1) = YI(I1)
      YP(I1) = YI(I1)
      ZC(I1,1) = ZI(I1)
      ZP(I1) = ZI(I1)
      XC(I) = XCENT(J) * CREF
      YC(I) = YCENT(J)
      ZC(I) = ZCFNT(J)
      AR(I) = AREA(J)
      IP = 4
      IBP = 1
```

SHIE 386
SHIE 387
SHIE 388
SHIE 389
SHIE 390
SHIE 391
SHIE 392
SHIE 393
SHIE 394
SHIE 395
SHIE 396
SHIE 397
SHIE 398
SHIE 399
SHIE 400
SHIE 401
SHIE 402
SHIE 403
SHIE 404
SHIE 405
SHIE 406
SHIE 407
SHIE 408
SHIE 409
SHIE 410
SHIE 411
SHIE 412
SHIE 413
SHIE 414
SHIE 415
SHIE 416
SHIE 417
SHIE 418
SHIE 419
SHIE 420

SHIELD

SHIELD

SHIE 421
 SHIE 422
 SHIE 423
 SHIE 424
 SHIE 425
 SHIE 426
 SHIE 427
 SHIE 428
 SHIE 429
 SHIE 430
 SHIE 431
 SHIE 432
 SHIE 433
 SHIE 434
 SHIE 435
 SHIE 436
 SHIE 437
 SHIE 438
 SHIE 439
 SHIE 440
 SHIE 441
 SHIE 442
 SHIE 443
 SHIE 444
 SHIE 445
 SHIE 446
 SHIE 447
 SHIE 448
 SHIE 449
 SHIE 450
 SHIE 451
 SHIE 452
 SHIE 453
 SHIE 454
 SHIE 455

```

    GO TO 750
  280 CONTINUE
  C CHECK IF FIRST FLEMENT COMPLETELY CONTAINS THE SECOND
    DO 290 I1=1,4
    IF (I1.EQ. IC) GO TO 290
    DO 289 I2=1,4
    IF (LS(I1,I2).GT. 0.0) GO TO 310
  289 CONTINUE
  290 CONTINUE
  C
  C THE FIRST ELEMENT COMPLETELY CONTAINS THE SECOND SINCE ALL L#S
  C WERE FOUND TO BE .LE. 0.0
  C SET UP AND SAVE THE NEW SHILED ELEMENT (IT IS THE SAME AS THE 2ND)
    DO 300 I1=1,4
    XP(I1) = ELEM(I1+10)
    YP(I1) = ELEM(I1+14)
    ZP(I1) = ELEM(I1+18)
    YPIE(I1) = YS(I1)
    ZPIE(I1) = ZS(I1)
  300 XP(5) = XP(1)
    YP(5) = YP(1)
    ZP(5) = ZP(1)
    YPIE(5) = YPIE(1)
    ZPIE(5) = ZPIE(1)
    IPF(1) = 10
    IPF(2) = 20
    IPF(3) = 30
    IPF(4) = 40
    IP = 6
    Y(5) = Y(1)
  
```

SHIELD

SHIELD

SHIE 456
 SHIE 457
 SHIE 458
 SHIE 459
 SHIE 460
 SHIE 461
 SHIE 462
 SHIE 463
 SHIE 464
 SHIE 465
 SHIE 466
 SHIE 467
 SHIE 468
 SHIE 469
 SHIE 470
 SHIE 471
 SHIE 472
 SHIE 473
 SHIE 474
 SHIE 475
 SHIE 476
 SHIE 477
 SHIE 478
 SHIE 479
 SHIE 480
 SHIE 481
 SHIE 482
 SHIE 483
 SHIE 484
 SHIE 485
 SHIE 486
 SHIE 487
 SHIE 488
 SHIE 489
 SHIE 490

```

Z(S) = Z(I)
XI(S) = XI(I)
YI(S) = YI(I)
ZI(S) = ZI(I)
YS(S) = YS(I)
ZS(S) = ZS(I)
LSH = L8H + 1
NE = 1
GO TO 630

C CHECK IF ELEMENTS 1 AND 2 WERE FOUND TO OVERLAP
310 IF (INT1(1) .FC. 0) GO TO 980

C ELEMENTS OVERLAP, FORM NEW NEGATIVE ELEMENTS FOR SHIELDED AREA
C**DETERMINE CORNERS OF POLYGON
C CHECK ALL L&S FOR POINTS INSIDE OF ELEMENT 1 AND SAVE ID NUMBERS
I3 = 0
DO 330 I2=1,4
DO 320 I1=1,4
IF (I1 .EQ. IC) GO TO 320
IF (LS(I1,I2) .GT. 0.0) GO TO 330
320 CONTINUE
I3 = I3 + 1
IV2(I3) = I2
330 CONTINUE

C CHECK FOR ANY ELEMENT 1 CORNERS THAT ARE INSIDE OF ELEMENT 2
IA = 0
DO 360 I1=1,4
IF (X1 .EQ. YC) GO TO 360
DO 350 I2=1,4
S2(1) = YS(I2+1) - YS(I2)
S2(2) = ZS(I2+1) - ZS(I2)
  
```

SHIELD

SHIELD

491 SHIE
 492 SHIE
 493 SHIE
 494 SHIE
 495 SHIE
 496 SHIE
 497 SHIE
 498 SHIE
 499 SHIE
 500 SHIE
 501 SHIE
 502 SHIE
 503 SHIE
 504 SHIE
 505 SHIE
 506 SHIE
 507 SHIE
 508 SHIE
 509 SHIE
 510 SHIE
 511 SHIE
 512 SHIE
 513 SHIE
 514 SHIE
 515 SHIE
 516 SHIE
 517 SHIE
 518 SHIE
 519 SHIE
 520 SHIE
 521 SHIE
 522 SHIE
 523 SHIE
 524 SHIE
 525 SHIE

IF (R2(1),EQ,0.0 .AND. S2(2).EQ,0.0) GO TO 350
 340 LS2(1) = (7(I1)-2S(I2))*S2(1) + (Y(I1)-Y8(I2))*S2(2)
 IF (LS2(1) .GT. 0.0) GO TO 360
 350 CONTINUE

C
 I4 = I4 + 1
 IV1(I4) = I1
 360 CONTINUE

C

IP = 0
 ICHK = 0
 IVT = IE + I3 + I4
 IF (IE.LE.2 .AND. IS.EQ.IVT) GO TO 980
 C CHECK IF ALL ELEMENT 2 CORNERS ARE OUTSIDE OF ELEMENT 1
 IF (I3 .EQ. 0) GO TO 430
 C SET ELEMENT 2 CORNER POINT AS POLYGON CORNER POINT
 I42 = IV2(1)

370 IP = IP + 1
 IPP(IP) = I42+10
 IF (IP .EQ. IVT) GO TO 540

C
 C CHECK IF THERE IS AN INTERSECTION ON THIS SIDE OF ELEMENT 2

380 DO 390 I1=1,IE
 IF (I1 .EQ. I11) GO TO 390
 IF (INT2(I1) .EQ. I42) GO TO 420

390 CONTINUE
 I42 = I42 + 1
 IF (I42 .GT. 4) I42 = 1
 IF (I3 .EQ. 0) GO TO 410
 DO 400 I1=1,I3
 IF (IV2(I1) .EQ. I42) GO TO 370

400 CONTINUE
 410 IF (ICMK .EQ. 1) GO TO 980
 IP = 0

SHIELD

SHIE 526
 SHIE 527
 SHIE 528
 SHIE 529
 SHIE 530
 SHIE 531
 SHIE 532
 SHIE 533
 SHIE 534
 SHIE 535
 SHIE 536
 SHIE 537
 SHIE 538
 SHIE 539
 SHIE 540
 SHIE 541
 SHIE 542
 SHIE 543
 SHIE 544
 SHIE 545
 SHIE 546
 SHIE 547
 SHIE 548
 SHIE 549
 SHIE 550
 SHIE 551
 SHIE 552
 SHIE 553
 SHIE 554
 SHIE 555
 SHIE 556
 SHIE 557
 SHIE 558
 SHIE 559
 SHIE 560

```

    ICHK = 1
    GO TO 430

C   SAVE INTERSECTION ON ELEMENT 2 SIDE AS NEXT POLYGON CORNER POINT
    420 IP = IP + 1
    IPP(IP) = I1
    IF (IP .EQ. IVT) GO TO 540
    I41 = I1
    GO TO 450

C   CHECK IF ANY ELEMENT 1 CORNERS ARE INSIDE OF ELEMENT 2
    430 IF (I4 .EQ. 0) GO TO 440
    I2 = 1
    I1 = 0
    GO TO 500

C   SET THE LOWEST ORDER INTERSECTION ON ELEMENT 1 AND SAVE AS CORNER
    440 IP = 1
    IPP(IP) = 1
    I41 = 1
    I1 = 1

C   CHECK IF THERE IS AN INTERSECTION ON THIS SIDE OF ELEMENT 1
    450 DO 460 I11=1,IF
        IF (I11 .EQ. I41) GO TO 460
        IF (INT1(I11) .EQ. INT1(I1)) GO TO 530
    460 CONTINUE

C   CHECK IF NEXT CORNER POINT OF ELEMENT 1 IS INSIDE OF ELEMENT 2
    470 IF (I4 .EQ. 0) GO TO 490
    IF (I1 .GT. IE) GO TO 490
    IF (I1 .EQ. 0) GO TO 980
    I41 = INT1(I1) + 1
    IF (I41 .GT. 4) I41 = 1
    IF (I41 .EQ. IC) I41 = I41 + 1
    IF (I41 .GT. 4) I41 = 1
  
```

SHIELD

```
DD 480 I2E1,I4
IF (I1V1(I2) .EQ. I41) GO TO 500
480 CONTINUE
490 I43 = I41
GO TO 510

C SAVE ELEMENT 1 CORNER AS NEXT POLYGON CORNER
500 IP = IP + 1
I43 = I1V1(I2)
IPF(IP) = I43*100
IF (IP .EQ. IVT) GO TO 540

C CHECK IF THE NEXT CORNER OF ELEMENT 1 IS ALSO INSIDE OF ELEMENT 2
I2 = I2 + 1
IF (I2 .GT. I4) GO TO 510
I44 = I43 + 1
IF (I1V1(I2) .EQ. I44) GO TO 500

C CHECK IF THERE IS AN INTERSECTION ON THIS SIDE OF ELEMENT 1
510 DD 520 I11=1,IE
IF (INT1(I11) .EQ. I43) GO TO 530
520 CONTINUE

C I41 = I41 + 1
GO TO 470

C SAVE INTERSECTION ON SIDE OF ELEMENT 1 AS NEXT POLYGON CORNER
530 IP = IP + 1
IPF(IP) = I11
IF (IP .EQ. IVT) GO TO 540
I42 = INT2(I11)
GO TO 380

C POLYGON HAS BEEN COMPLETED. IT HAS IP CORNER POINTS.
C
```

```
SHIE 561
SHIE 562
SHIE 563
SHIE 564
SHIE 565
SHIE 566
SHIE 567
SHIE 568
SHIE 569
SHIE 570
SHIE 571
SHIE 572
SHIE 573
SHIE 574
SHIE 575
SHIE 576
SHIE 577
SHIE 578
SHIE 579
SHIE 580
SHIE 581
SHIE 582
SHIE 583
SHIE 584
SHIE 585
SHIE 586
SHIE 587
SHIE 588
SHIE 589
SHIE 590
SHIE 591
SHIE 592
SHIE 593
SHIE 594
SHIE 595
```

SHIELD

SHIELD

```
540 IF (IP .GE. 3) GO TO 550
C WRITE (6,522) J,K,X,Y,Z,YS,ZS,LS
C 522 FOPMAT (1H0,48H**THERE ARE NOT ENOUGH INTERSECTIONS AND INSIDE,
C 136H CORNER POINTS TO FORM A NEW ELEMENT,/1H ,2X,2HJ$,13,
C 2 4H K$,13,/, (3X,5F10,4)
C GO TO 580
C
C * SET UP POLYGON CORNER POINTS
550 DO 620 IE=1,IP
C CHECK IF INTERSECTION OF SIDES, CORNER OF ELEMENT 2 OR FLEMENT 1
IF (TPF(IE) .LT. 10) GO TO 560
IF (ZPF(IE) .LT. 100) GO TO 600
GO TO 610
C
C STDE INTERSECTION
560 I7 = IPF(IE)
I1 = INT1(I7)
I2 = INT2(I7)
SAS = 1.0 / (1.0 + ABS(LS(I1,I2+1))/LS(I1,I2))
YPIE(IE) = SAS * (YS(I2+1)-YS(I2)) + YS(I2)
ZPIE(IE) = SAS * (ZS(I2+1)-ZS(I2)) + ZS(I2)
DV = ARS(Y(I1+1)-Y(I1))
DZ = ARS(Z(I1+1)-Z(I1))
IF (DZ .GT. DY) GO TO 580
IF (DY .GT. 0.00001) GO TO 570
DAD = 0.0
GO TO 590
570 DAD = ABS((YPIE(IE)-Y(I1))/DY)
GO TO 590
580 DAD = ARS((ZPIE(IE)-Z(I1))/DZ)
590 XP(IE) = XI(I1) + (XI(I1+1)-XI(I1)) * DAD
YP(IE) = YI(I1) + (YI(I1+1)-YI(I1)) * DAD
ZP(IE) = ZI(I1) + (ZI(I1+1)-ZI(I1)) * DAD
GO TO 620
C
```

SHIE 596
SHIE 597
SHIE 598
SHIE 599
SHIE 600
SHIE 601
SHIE 602
SHIE 603
SHIE 604
SHIE 605
SHIE 606
SHIE 607
SHIE 608
SHIE 609
SHIE 610
SHIE 611
SHIE 612
SHIE 613
SHIE 614
SHIE 615
SHIE 616
SHIE 617
SHIE 618
SHIE 619
SHIE 620
SHIE 621
SHIE 622
SHIE 623
SHIE 624
SHIE 625
SHIE 626
SHIE 627
SHIE 628
SHIE 629
SHIE 630

SHIELD

SHIELD

```
C CORNER POINT OF ELEMENT 2
600 I7 = IPF(IE)/10
XP(IE) = ELEM(I7+10)
YP(IE) = ELEM(I7+14)
ZP(IE) = ELEM(I7+18)
GO TO 620

C CORNER POINT OF ELEMENT 1
610 I7 = IPF(IE)/100
XP(IE) = XI(I7)
YP(IE) = YI(I7)
ZP(IE) = ZI(I7)

C
620 CONTINUE
630 CONTINUE
C **PROJECTION OF ELEMENT 2 CORNER POINTS INTO PLANE OF INPUT ELEMENT*
C
IPF(IP+1) = IPF(1)
DO 740 IE=1,IP
C CHECK IF THIS POLYGON CORNER PT. WAS AN ELEMENT 2 CORNER
IF (IPF(IE).LT.1.0 .OR. IPF(IE).GT.40) GO TO 740
IEE = IE + 1
I7 = 0
I2 = IPF(IE)/10
C CALCULATE SIDE VECTORS
640 S2(1) = YS(I2+1) - YS(I2)
S2(2) = ZS(I2+1) - ZS(I2)
CHECK IF ZERO LENGTH SIDE
IF (S2(1).EQ.0.0 .AND. S2(2).EQ.0.0) GO TO 650
GO TO 660
650 I2 = I2 + 1
IF (I2.EQ. 5) I2 = 1
GO TO 640
C SEARCH FOR SIDE PROJECTION INTERSECTIONS AND CALCULATE COORDINATES
660 DO 690 I1=1,4
```

SHIE 631
SHIE 632
SHIE 633
SHIE 634
SHIE 635
SHIE 636
SHIE 637
SHIE 638
SHIE 639
SHIE 640
SHIE 641
SHIE 642
SHIE 643
SHIE 644
SHIE 645
SHIE 646
SHIE 647
SHIE 648
SHIE 649
SHIE 650
SHIE 651
SHIE 652
SHIE 653
SHIE 654
SHIE 655
SHIE 656
SHIE 657
SHIE 658
SHIE 659
SHIE 660
SHIE 661
SHIE 662
SHIE 663
SHIE 664
SHIE 665

SHIELD

```

LS2(1) = (Z(I1)-ZS(I2))*S2(1) - (Y(I1)-YS(I2))*S2(2)
LS2(2) = (Z(I1+1)-ZS(I2))*S2(1) - (Y(I1+1)-YS(I2))*S2(2)
IF (LS2(1).GT.0.0 .AND. LS2(2).GT.0.0) GO TO 690
IF (LS2(1).LT.0.0 .AND. LS2(2).LT.0.0) GO TO 690
I7 = I7 + 1
IF (ABS(LS2(1)) .GT. 0.00001) GO TO 670
XSID(I7) = XI(I1)
YSID(I7) = YI(I1)
ZSID(I7) = ZI(I1)
YSIP(I7) = YI(I1)
ZSIP(I7) = ZI(I1)
GO TO 680
670 SAS = 1.0 / (1.0+ABS(LS2(2)/LS2(1)))
C CALCULATE SIDE INTERSECTION IN INPUT COORDINATE SYSTEM
XSID(I7) = XI(I1) + SAS*(XI(I1+1)-XI(I1))
YSID(I7) = YI(I1) + SAS*(YI(I1+1)-YI(I1))
ZSID(I7) = ZI(I1) + SAS*(ZI(I1+1)-ZI(I1))
C CALCULATE SIDE INTERSECTION COORDINATES IN VIEWING REFERENCE COORD,
YSIP(I7) = YI(I1) + SAS*(Y(I1+1)-Y(I1))
ZSIP(I7) = ZI(I1) + SAS*(Z(I1+1)-Z(I1))
680 CONTINUE
IF (I7 .EQ. 2) GO TO 700
690 CONTINUE
C CALCULATE COORDINATES OF CORNER PT. IN ELEMENT PLANE
700 DY = ABS(YSIP(2))-YSIP(1)
DZ = ABS(ZSIP(2))-ZSIP(1)
IF (DZ .GT. DY) GO TO 720
IF (DY .GT. 0.00001) GO TO 710
DAD = 0.0
GO TO 730
710 DAD = ABS((YS(I2)-YSIP(1))/DY)
GO TO 730
720 DAD = ABS((ZS(I2)-ZSIP(1))/DZ)
730 XP(IF) = XSID(1) + (XSID(2)-XSID(1))*DAD
YP(IF) = YSID(1) + (YSID(2)-YSID(1))*DAD

```

```

SHIE 666
SHIE 667
SHIE 668
SHIE 669
SHIE 670
SHIE 671
SHIE 672
SHIE 673
SHIE 674
SHIE 675
SHIE 676
SHIE 677
SHIE 678
SHIE 679
SHIE 680
SHIE 681
SHIE 682
SHIE 683
SHIE 684
SHIE 685
SHIE 686
SHIE 687
SHIE 688
SHIE 689
SHIE 690
SHIE 691
SHIE 692
SHIE 693
SHIE 694
SHIE 695
SHIE 696
SHIE 697
SHIE 698
SHIE 699
SHIE 700

```

SHIELD

```

ZP(IE) = ZSID(1) + (ZSID(2)-ZSID(1))*DAD
740 CONTINUE
750 CONTINUE
C POLYGON CORNER POINTS ARE NOW COMPLETE AND STORED IN XP, YP, ZP
C MAKE CHECK FOR DEFINITE SHIELDING
  XMAX = AMAX1(X(1),X(2),X(3),X(4))
  IF (IP .EQ. 0) GO TO 770
  DO 760 IE=IP,7
    XP(IE+1) = 99999.9
760 XMIN2 = AMIN1 (XP(1),XP(2),XP(3),XP(4),XP(5),XP(6),XP(7),XP(8))
770 XMIN2 = GF. XMAX) GO TO 790
  IF (XMIN2 .GT. XMAX) GO TO 790
C SHIELDING NOT DEFINITE, SO CHECK FURTHER
  DO 780 IF=1,IP
  IF (IPF(IE).GT.9 .AND. IPF(IE).LT.50) GO TO 780
  NX2 = YP(IE) * ELEM(7)
  NY2 = YP(IE) * ELEM(8)
  NZ2 = ZP(IE) * ELEM(9)
  VN = SQRT (NX2**2 + NY2**2 + NZ2**2)
  IF (VN .EQ. 0.0) GO TO 780
  NX2 = NX2/VN
  NY2 = NY2/VN
  NZ2 = NZ2/VN
  NDN2 = ELEM(4)*NX2 + ELEM(5)*NY2 + ELEM(6)*NZ2
  IF (NDN2 .GT. 0.0001) GO TO 980
780 CONTINUE
  IF (IBP .EQ. 1) GO TO 950
790 CONTINUE
C SET UP PROJECTION TRANSFORMATION DATA FOR SHIELDED ELEMENT
  T1X = XI(3) * XI(1)
  T2X = XI(4) * XI(2)
  T1Y = YI(3) * YI(1)
  T2Y = YI(4) * YI(2)
  T1Z = ZI(3) * ZI(1)
  T2Z = ZI(4) * ZI(2)

```

SMIE 701
SMIE 702
SMIE 703
SMIE 704
SMIE 705
SMIE 706
SMIE 707
SMIE 708
SMIE 709
SMIE 710
SMIE 711
SMIE 712
SMIE 713
SMIE 714
SMIE 715
SMIE 716
SMIE 717
SMIE 718
SMIE 719
SMIE 720
SMIE 721
SMIE 722
SMIE 723
SMIE 724
SMIE 725
SMIE 726
SMIE 727
SMIE 728
SMIE 729
SMIE 730
SMIE 731
SMIE 732
SMIE 733
SMIE 734
SMIE 735

SHIELD

SHIELD

SMIE	736
SMIE	737
SMIE	738
SMIE	739
SMIE	740
SMIE	741
SMIE	742
SMIE	743
SMIE	744
SMIE	745
SMIE	746
SMIE	747
SMIE	748
SMIE	749
SMIE	750
SMIE	751
SMIE	752
SMIE	753
SMIE	754
SMIE	755
SMIE	756
SMIE	757
SMIE	758
SMIE	759
SMIE	760
SMIE	761
SMIE	762
SMIE	763
SMIE	764
SMIE	765
SMIE	766
SMIE	767
SMIE	768
SMIE	769
SMIE	770

```

AVX = 0.25*(XI(1)+XI(2)+XI(3)+XI(4))
AVY = 0.25*(YI(1)+YI(2)+YI(3)+YI(4))
AVZ = 0.25*(ZI(1)+ZI(2)+ZI(3)+ZI(4))
T = SQRT(TIX*TIX + TIY*TIY + TIZ*TIZ)
IF (T .EQ. 0.0) GO TO 800
TIX = TIX/T
TIY = TIY/T
TIZ = TIZ/T
R00 T2X = NY(J)*T17*CREF - NZ(J)*TIY
T2Y = NZ(J)*TIX - NX(J)*TIZ
T2Z = NX(J)*TIY - NY(J)*T1X*CREF
C PROJECT POLYGON POINTS INTO QUADRILATERAL PLANE
DO 810 IE=1,JP
IF (IPF(IE) .GT. 40) GO TO 810
XN = T1X*(XP(IE)-AVX) + TIY*(YP(IE)-AVY) + TIZ*(ZP(IE)-AVZ)
YN = T2X*(XP(IE)-AVX) + T2Y*(YP(IE)-AVY) + T2Z*(ZP(IE)-AVZ)
XP(IE) = AVX + T1X*XN + T2X*YN
YP(IE) = AVY + T1Y*XN + T2Y*YN
ZP(IE) = AVZ + T1Z*XN + T2Z*YN
R10 CONTINUE
C CONVERT POLYGON POINTS TO THE REQUIRED NUMBER OF ELEMENTS
DO 820 II=1,4
XC(II,1) = XP(II)
YC(II,1) = YP(II)
R20 ZC(II,1) = ZP(II)
NE = 1
IF (IP .NE. 3) GO TO 830
XC(4,1) = XP(3)
YC(4,1) = YP(3)
ZC(4,1) = ZP(3)
GO TO 880
R30 IF (YP .EQ. 4) GO TO 880
DD 840 II=1,4

```

SHIELD

SHIELD

SHIE	771
SHIE	772
SHIE	773
SHIE	774
SHIE	775
SHIE	776
SHIE	777
SHIE	778
SHIE	779
SHIE	780
SHIE	781
SHIE	782
SHIE	783
SHIE	784
SHIE	785
SHIE	786
SHIE	787
SHIE	788
SHIE	789
SHIE	790
SHIE	791
SHIE	792
SHIE	793
SHIE	794
SHIE	795
SHIE	796
SHIE	797
SHIE	798
SHIE	799
SHIE	800
SHIE	801
SHIE	802
SHIE	803
SHIE	804
SHIE	805

```

      XO(I1,2) = XP(I1+3)
      YO(I1,2) = YP(I1+3)
      ZO(I1,2) = ZP(I1+3)
      NE = 2
      IF (IP, NE, 5) GO TO 850
      XO(3,2) = XP(1)
      YO(3,2) = YP(1)
      ZO(3,2) = ZP(1)
      XO(4,2) = XP(1)
      YO(4,2) = YP(1)
      ZO(4,2) = ZP(1)
      GO TO 850
250 IF (IP, NE, 6) GO TO 860
      YO(4,2) = XP(1)
      YO(4,2) = YP(1)
      ZO(4,2) = ZP(1)
      GO TO 850
      C
      XO(1,3) = XP(4)
      YO(1,3) = YP(4)
      ZO(1,3) = ZP(4)
      XO(2,3) = XP(7)
      YO(2,3) = YP(7)
      ZO(2,3) = ZP(7)
      NE = 3
      IF (IP, NE, 7) GO TO 870
      XO(3,3) = XP(1)
      YO(3,3) = YP(1)
      ZO(3,3) = ZP(1)
      XO(4,3) = XP(1)
      YO(4,3) = YP(1)
      ZO(4,3) = ZP(1)
      GO TO 830
      A70 XO(3,3) = XP(8)
      YO(3,3) = YP(8)

```

SHIELD

SHIELD

SHIE 006
 SHIE 007
 SHIE 008
 SHIE 009
 SHIE 010
 SHIE 011
 SHIE 012
 SHIE 013
 SHIE 014
 SHIE 015
 SHIE 016
 SHIE 017
 SHIE 018
 SHIE 019
 SHIE 020
 SHIE 021
 SHIE 022
 SHIE 023
 SHIE 024
 SHIE 025
 SHIE 026
 SHIE 027
 SHIE 028
 SHIE 029
 SHIE 030
 SHIE 031
 SHIE 032
 SHIE 033
 SHIE 034
 SHIE 035
 SHIE 036
 SHIE 037
 SHIE 038
 SHIE 039
 SHIE 040

```

    ZC(3,3) = ZP(8)
    XC(4,3) = XP(1)
    YO(4,3) = YP(1)
    ZC(4,3) = ZP(1)
    GO TO 880

  C 880 CONTINUE
  C CALCULATE AREA AND CENTROID FOR NEW NEGATIVE ELEMENTS
  DO 940 I=1,NE
    T1X = XC(3,I) - XC(1,I)
    T2X = XC(4,I) - XC(2,I)
    T1Y = YC(3,I) - YC(1,I)
    T2Y = YC(4,I) - YC(2,I)
    T1Z = ZC(3,I) - ZC(1,I)
    T2Z = ZC(4,I) - ZC(2,I)
    AVX = 0.25*(XC(1,I)+XC(2,I)+XC(3,I)+XC(4,I))
    AVY = 0.25*(YC(1,I)+YC(2,I)+YC(3,I)+YC(4,I))
    AVZ = 0.25*(ZC(1,I)+ZC(2,I)+ZC(3,I)+ZC(4,I))
    D = NX(J)*(AVX-XC(1,I)) + NY(J)*(AVY-YC(1,I))+CREF +
      1  NZ(J)*(AVZ-ZC(1,I))
    PD = ABS(D)
    T = SQRT(T1X*T1X + T1Y*T1Y + T1Z*T1Z)
    IF (T.LT. 0.0001) GO TO 890
    T1X = T1X/T
    T1Y = T1Y/T
    T1Z = T1Z/T
    T2X = NY(J)*T1Z+CREF - NZ(J)*T1Y
    T2Y = NZ(J)*T1X - NX(J)*T1Z
    T2Z = NX(J)*T1Y - NY(J)*T1X+CREF
    ON 900 I=1,4
    XPA(I2) = XC(I2,I) + NX(J)*D
    YPA(I2) = YC(I2,I) + NY(J)*D+CREF
    ZPA(I2) = ZC(I2,I) + NZ(J)*D
    D = - D
    XPIF = XPA(I2) - AVX
  
```

SHIELD

SHIELD

SHIE	861
SHIE	862
SHIE	863
SHIE	864
SHIE	865
SHIE	866
SHIE	867
SHIE	868
SHIE	869
SHIE	870
SHIE	871
SHIE	872
SHIE	873
SHIE	874
SHIE	875

```

YDIF = YPA(I2) - AVY
ZDIF = ZPA(I2) - AVZ
XII(I2) = T1X*XDIF + T1Y*YDIF + T1Z*ZDIF
ETA(I2) = T2X*XDIF + T2Y*YDIF + T2Z*ZDIF
ETACK = ETA(2) - ETA(4)
IF (ABS(ETACK) .GT. 0.0001) GO TO 910
XIO = 0.0
GO TO 920
910 XIO = 0.333333*(XII(I2)*(ETA(1)-ETA(2)) + XII(2)
      *(ETA(4)-ETA(1))) / (ETA(2)-ETA(4))
920 ETAO = -0.333333 * ETA(1)
      DO 930 I2=1,4
930 XII(I2) = XII(I2) + XIO
      ETA(I2) = ETA(I2) - ETAO
      XC(I1) = AVX + T1X*XIO + T2X*ETAO
      YC(I1) = AVY + T1Y*XIO + T2Y*ETAO
      ZC(I1) = AVZ + T1Z*XIO + T2Z*ETAO
      XT3M1 = XII(3) - XII(1)
      ETA2M4 = ETA(2) - ETA(4)
      AR(I1) = 0.5*XI3M1*ETA2M4
940 CONTINUE
950 CONTINUE
C
DO 970 I6=1,NE
IF (AR(I6) .LE. 0.0001) GO TO 970
ICT = ICT + 1
E(1) = ICT
E(2) = IN(-)
E(3) = IM(J)
E(4) = NX(J)
E(5) = NY(J) * CREF
E(6) = NZ(J)
E(7) = XC(I6)
E(8) = YC(I6)
E(9) = ZC(I6)

```

SHIELD

SHIELD

SMIE	876
SMIE	877
SMIE	878
SMIE	879
SMIE	880
SMIE	881
SMIF	882
SMIE	883
SMIE	884
SMIE	885
SMIE	886
SMIE	887
SMIE	888
SMIE	889
SMIE	890
SMIE	891
SMIE	892
SMIE	893
SMIE	894
SMIE	895
SMIE	896
SMIE	897
SMIE	898
SMIE	899
SMIE	900
SMIE	901
SMIE	902
SMIE	903
SMIE	904
SMIE	905
SMIE	906
SMIE	907
SMIE	908
SMIE	909
SMIE	910

SHIELD

```

E(10) = -AR(I8)
E(11) = XC(1,I8)
E(12) = XO(2,I8)
E(13) = XO(3,I8)
E(14) = XO(4,I8)
E(15) = YO(1,I8)
E(16) = YO(2,I8)
E(17) = YO(3,I8)
E(18) = YO(4,I8)
E(19) = ZO(1,I8)
E(20) = ZO(2,I8)
E(21) = ZO(3,I8)
E(22) = ZO(4,I8)
E(23) = 4.0
E(24) = 0.0
E(25) = 0.0
WRITE (TAPE4) F
IF (IPRINT .NE. 0) WRITE (TAPE0T,960) F
960 FORMAT (1H0,8X,3F4.1,2X,7E14.5,1H ,22X,4E14.5,1H ,22X,4E14.5,1
1 1H ,22X,4E14.5,1H ,22X,3E14.5)
970 CONTINUE
980 IF (IREF2.EQ.2 .OR. ISMY.EQ.1 .OR. BETA(I).EQ.0.0) GO TO 990
IREF2 = 2
CREF2 = -1.0
GO TO 140
C
990 CONTINUE
1000 IF (IREF1.EQ.2 .OR. ISMY.EQ.1 .OR. BETA(I).EQ.0.0) GO TO 1010
IREF1 = 2
CREF = -1.0
GO TO 120
1010 CONTINUE
C
C
I9 = I9 + 1

```

SHIELD

SHIE 911
SHIE 912
SHIE 913
SHIE 914
SHIE 915
SHIE 916
SHIE 917
SHIE 918
SHIE 919
SHIE 920
SHIE 921
SHIE 922
SHIE 923
SHIE 924
SHIE 925I
SHIE 926C
SHIE 927

```
NSHE(19) = ICT
1020 CONTINUE
C END OF ALPHA-BETA LOOP
C
C CHECK FOR END OF COMPONENT ANALYSIS LOOP
  IF (NPAW .LT. NPAWL) GO TO 30
  REWIND TAPE
  GO TO 1050
C
1030 WRITE (6,1040) ISIZE
1040 FORMAT (1H0,43H**ERROR**NUMBER OF ELEMENTS IN A COMPONENT,
1 13HCANNOT EXCEED ,14,34H A STOP WILL BE CALLED IN ROUTINE ,
2 8H SHIELD, )
      STOP
C1050 RETURN
1050 CONTINUE
      END
```

SHIELD

PRES

001C
 002C
 003I
 004
 005
 006
 007
 008
 009
 010
 011
 012
 013
 014
 015
 016
 017
 018
 019
 020
 021
 022
 023
 024
 025
 026
 027
 028
 029
 030
 031
 032
 033
 034
 035

```

OVERLAY (MARK4,2,3)
PROGRAM PRES
SURROUTINE PRES

C THIS IS THE EXECUTIVE ROUTINE FOR THE INVISCID PRESSURE CALCULATIONS
C
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /GDATA/LTOT, J, SYMFCT, JORN, IGTYPF, I
COMMON /FLIGHT/MACH, ALT, PSTAG, TSTAG, V, PEND, PFS, TFS, AFS, RHOF5, VIS
COMMON /TAPE/TAPFIN, TAPEIN, TAPEA, TAPEB, TAPEC, TAPEF, TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 COMMON /ABDATA/NAB, ALPHA(20), BETA(20), ROL(20), CDELTA(20), QI(20),
2 HI(20), PI(20)
3 COMMON /REF/SRFF, MAC, SPAN, XCG, YCG, ZCG
COMMON /INTERF/INT, ISHE(20), NSHE(100), T9, ISHEF, INF(20,22),
1 DINF(20,6), LNTAB(20)
COMMON /FARRAY/F
COMMON /MDATA/IMPACT, ISHAD, IPRJNT, PDATA, TSAVE, IPRJN
COMMON /GASD/GAM, GASCP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
COMMON /SURFENP/INORM, ISURF, IX, IY, IZ, UI, UJ, UL, VI, VL, CR, CT, CHX, N2, N3,
1 XO, YO, ZO, AP11, AP12, AP13, AP21, AP22, AP23, AP31, AP32, AP33,
2 FLOWC(7), XI(500), YI(500), R(503,7), XKF(503), AA(5990),
3 XR(4), YB(4), ZB(4),
4 NOSET, IABSET, IR, LORG(20), ISR(20), M3, TSFR, IVIS, IFTYP, IFLOW,
5 RX(2,20), RY(2,20), NR
DIMENSION F(15), E(25), FP(25)
DIMENSION F(11), IPANL(10), PDATA(6), IPR(20), FF(11), IMP(20), ISA(20),
1 ISH(20), PDA(6,20), COM(10)
DIMENSION LCOM(41), LSUM(41), TITL(5)
DIMENSION INFT(22), DINF(6), E1(5), EP(11), E4(41), E5(41), E6(41),
1 E7(11), E8(14)
DIMENSION IG4S(1000)
DIMENSION IMP(20)
INTEGER FROR, SYMFCT, PAGE, CASE
INTEGER TAPFIN, TAPEIN, TAPEA, TAPEB, TAPEC, TAPEF, TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK, TAPEF, TAPEF,
  
```

PRES

PRES

```

1      TAPFG,TAPEH,TAPEI,TAPFJ,TAPEK
   REAL  MACH,MAC
   DATA ISIZE/1000/
   DATA TITLE9/10HFORCE DATA,10M SAVE UNIT,10H
1 10H      /
   DATA YTITLE9/4HFORC,4HE DA,4HTA 8,6HAVE ,4HUNIT/
C
C
C
C SFT INITIAL CONSTANTS FOR START OF CASE
   ERROR = 0
   NCOM = 0
   I9 = 0
C
   WRITE (TAPECT,10)
10  FORMAT (1H1,32H*** PRESSURE CALCULATION PROGRAM )
C
C READ YTITLE CARD AND FLAGS
   READ (TAPEIN,20) NCOMP,IFSAVE,TITLE
20  FORMAT (12,11,3X,15A4)
C SFT UP CONSTANTS FOR FORCE DATA SAVE ON UNIT 9
   IF (IFSAVE .EQ. 2) GO TO 60
   IF (IFSAVE .NE. 0) GO TO 40
C
C
C INITIALIZE CONSTANTS FOR UNIT 9 FORCE DATA SAVE UNIT
   NTOT = TOTAL NUMBER OF COMPONENTS SAVED
   ISUM = TOTAL NUMBER OF SUMMATION SETS
   NEXT = NEXT EMPTY RECORD ON UNIT 9
   LCOM = POINTERS TO COMPONENT DATA (01) 1ST RECORD IS DATA TYPE
   LSUM = POINTERS TO SUMMATION DATA (01) 1ST RECORD IS DATA TYPE
           (DATA TYPE=1 FORCE ONLY, =2 FORCE + STABILITY DATA)
C
   NTOT = 0
   ISUM = 0
   NEXT = 10
   DO 21 IS1,41

```

PRES

PRES

PRES 071
PRES 072
PRES 073
PRES 074
PRES 075
PRES 076
PRES 077C
PRES 078I
PRES 079
PRES 080
PRES 081
PRES 082
PRES 083
PRES 084
PRES 085
PRES 086
PRES 087
PRES 088
PRES 089C
PRES 090I
PRES 091
PRES 092
PRES 093
PRES 094
PRES 095
PRES 096
PRES 097
PRES 098
PRES 099
PRES 100
PRES 101
PRES 102
PRES 103
PRES 104
PRES 105

PRES

```
      LCOM(I) = 0
21  LSUM(I) = 0
      GO TO 50
C
C  RETRIEVE PREVIOUSLY SAVED TABLE OF CONTENTS
40  I99 = 1
      CALL READMS (9,E1,8,I99)
      READ (2*I99) E1
      NTGT = E1(1)
      ISUM = E1(2)
      NEXT = E1(3)
C
C  CHECK IF VALID RECORD
      IF (F1(4) .EQ. TITLE9(1)) GO TO 43
      WRITE (TAPEOUT,42)
42  FORMAT (1H0,4I10**IFSAVE WAS INPUT = 1 IN PRES YET UNIT 9 HAS ,
           1 55HNEVER BEEN INITIALIZED WITH A IFSAVE = 0. PROGRAM STOP. )
      STOP
43  I99 = 2
      CALL READMS (9,E5,41,I99)
      READ (9*I99) E5
      DO 45 I=1,40
45  LCOM(I) = E5(I+1)
C
C
50  DO 22 I=1,20
      EQ(I+1) = ALPHA(I)
22  EQ(I+21) = BETA (I)
      EQ(I) = NAB
      DO 24 I=1,5
24  E1(I+3) = TITLE9(I)
25  E6(I) = 0.0
      DO 26 I=1,11
26  E7(I) = 0.0
C
C  READ COMPONENT SELECTION CARD + INTERFERENCE FLAGS
```

PRES

```

90 READ (TAPEIN,70) IPANL,IPM,INT,ISHEF
70 FORMAT (10I2,3I1)
NCCM = NCOM + 1
C *****
C CHECK IF NEW PRESSURE DATA ARE TO BE READ
IF (IPM.GT. 1) GO TO 110
C READ PRESSURE METHOD DATA (1 CARD OR NAB CARDS)
DO 100 I=1,NAB
IF (IPM.FG.0 .OR. I.EQ.1) READ (TAPEIN,80) IMPACT,ISHAD,IPRINT,
1 IPIN,ISAVE,PDATA
80 FORMAT(2I2,3I1,3X,6F10.0)
IMP(I) = IMPACT
ISH(I) = ISHAD
IPR(I) = IPRINT
ISA(I) = ISAVE
INP(I) = IPIN
C
DO 90 K=1,6
90 PDA(K,I) = PDATA(K)
109 CONTINUE
110 CONTINUE
C *****
C SET UP INTERFERENCE DATA ARRAY (DINF)
INTF = 0
IF (INT.EQ.0 .OR. INT.EQ.5) GO TO 119
DO 115 I=1,NAB
IF (I.FG.1 .OR. INT.EQ.1)
111 READ(TAPEIN, 111) INFI
DO 113 J=1,2P
113 INF(I,J) = I
IF((I.EQ.1) .AND. INT.EQ.1) .AND. INFI(1).EQ.0)
112 READ(TAPEIN, 112) DINF

```

PRES

PRES

PRES 141
 PAFS 142
 PAFS 143
 PAFS 144
 PAFS 145
 PAFS 146
 PAFS 147
 PAFS 148
 PAFS 149
 PAFS 150
 PAFS 151
 PAFS 152
 PAFS 153
 PAFS 154
 PAFS 155
 PAFS 156
 PAFS 157C
 PAFS 158I
 PAFS :5C
 PAFS 160
 PAFS 161
 PAFS 162
 PAFS 163
 PAFS 164
 PAFS 165
 PAFS 166
 PAFS 167
 PAFS 168
 PAFS 169C
 PAFS 170I
 PAFS 171
 PAFS 172
 PAFS 173
 PAFS 174
 PAFS 175

```

112 FORMAT(10X, 6F10.0)
DO 114 J=1,6
114 CINF(I,J) = CINF(I,J)
115 CONTINUE
119 CONTINUE
C CHECK IF INTERFERENCE FLAG IS TO BE TURNED ON.
IF (INT .EQ. 0) GO TO 118
DO 117 I=1,NAB
117 CINF(I,1) .GT. 0) INTF = 1
118 CONTINUE
119 CONTINUE
IF (INTF .EQ. 1) REWIND TAPEF

C
C OBTAIN GEOMETRY DATA FROM SAVE UNIT FOR COMPLETE COMPONENT AND
C PLACE IN GEOMETRY DATA WORKING ARRAYS
IGA = 2
CALL READMS (4,E,25,IG4)
READ (4#IG4) E
NEXTG = E(1)
NP = E(2)
NPMAX = E(3)
NREM = E(4)
K = 0
NCS = 0
DO 130 I=1,10
1F (TPANL(I) .EQ. 0) GO TO 14C
NCS = NCS + 1
IG4 = IPANL(I)*5
CALL READMS (4,EP,25,IG4)
READ (4#IG4) EP
1STAT3 = EP(1)
COM(I) = EP(2)
1START = EP(3)
N = EP(4)
ICRN = EP(5)
  
```

PRES

PRES

PRES 176
PRES 177
PRES 178
PRES 179
PRES 180
PRES 181
PRES 182
PRES 183
PRES 184
PRES 185C
PRES 186I
PRES 187
PRES 188
PRES 189
PRES 190
PRES 191
PRES 192
PRES 193
PRES 194
PRES 195
PRES 196
PRES 197
PRES 198
PRES 199
PRES 200
PRES 201
PRES 202
PRES 203
PRES 204
PRES 205
PRES 206
PRES 207
PRES 208
PRES 209
PRES 210

PRES

```
C      SYMFCY = EP(6)
      ISTART = ISTART + NREM
      DO 120 II=1,N
      K = K + 1
      IF (K .GT. ISIZE) GO TO 150
      IG4 = ISTART + NREM*II
      IG4S(K) = IG4
      IF (INTF .NE. 1) GO TO 120
      CALL READMS (4,ELEM,25,IG4)
      READ (4,IG4) ELEM
      WRITE (TAPEE) IG4,ELEM
120 CONTINUE
130 CONTINUE
C
140 LTOT = K
      GO TO 170
150 WRITE (TAPEOT,160) ISIZE,ISIZE
160 FORMAT (1H ,46H*** ERROR ** NUMBER OF FLPMFNTS PER COMPONENT ,
1      13HCANNON EXCEED ,14,/1H ,14X,22HNDIVIDE THE SHAPE INTO ,
2      47HMORE COMPONENTS WITH NO COMPONENT GREATER THAN ,14,
3      9H ELEMENTS )
      STOP
170 CONTINUE
C
      ISRF = 0
      DO 172 I = 1,NAB
      IF (YMP(I) .GT. 0) ISRF = 1
      :72 CONTINUE
      IF (ISRF .EQ. 0) GO TO 175
C
C ELEMENTS MUST BE WRITTEN ON TAPEE
C CHECK IF ALREADY DONE FOR INTERFERENCE FLOWS
      IF (INTF .EQ. 1) GO TO 175
C
```

PRES

```
DO 174 I = 1,LTOT
IG4 = IG4S(I)
CALL READMS(4,ELEM,25,IG4)
READ(4,IG4) ELEM
WRITE(TAPEF) IG4,FLEM
174 CONTINUE
175 CONTINUE
C
C
C CHECK FOR INTERFERENCE USAGE
IF (INTF, EQ, 0) GO TO 181
C CALL FLOW FIELD PRE-PROCESSOR FOR INTERFERENCE DATA
CALL FLNTRP
C
C
C 181 CONTINUE
C
C CHECK FOR INPUT SURFACE PRESSURES
IF (ISRF, EQ, 0) GO TO 183
C CALL PRE-PROCESSOR FOR SURFACE DATA
CALL CPINPT(INP, LTOTAB)
183 CONTINUE
C
C
C GEOMETRY DATA LOAD IS COMPLETE
ISAS = 0
IF (IFSAVE, EQ, 2) GO TO 182
NTOT = NTOT + 1
LCOM(NTOT) = NEXT
```

PRES 211
PRES 212
PRES 213C
PRES 214I
PRES 215
PRES 216
PRES 217
PRES 218
PRES 219
PRES 220
PRES 221
PRES 222
PRES 223
PRES 224
PRES 225
PRES 226
PRES 227
PRES 228
PRES 229
PRES 230
PRES 231
PRES 232
PRES 233
PRES 234
PRES 235
PRES 236
PRES 237
PRES 238
PRES 239
PRES 240
PRES 241
PRES 242
PRES 243
PRES 244
PRES 245

PRES

PRES

PRES 246
 PRES 247
 PRES 248
 PRES 249
 PRES 250
 PRES 251
 PRES 252
 PRES 253
 PRES 254
 PRES 255
 PRES 256
 PRES 257
 PRES 258
 PRES 259
 PRES 260
 PRES 261
 PRES 262
 PRES 263
 PRES 264
 PRES 265
 PRES 266C
 PRES 267I
 PRES 268
 PRES 269C
 PRES 270I
 PRES 271
 PRES 272
 PRES 273
 PRES 274
 PRES 275
 PRES 276
 PRES 277
 PRES 278
 PRES 279
 PRES 280

PRES

```

NEXT = NEXT + 2
182 CONTINUE
C
C*****
C START ALPHA=BETA LOOP (DO ALL ALPHA=BETAS FOR GIVEN COMPONENT)
  DO 190 J=1,MA8
    IMPACT = IMP(J)
    ISHAD = ISH(J)
    IPRINT = IPR(J)
    ISAVE = ISA(J)
    IPIH = INP(J)
    IF (ISAVE .EQ. 1) ISAS = 1
    DO 180 I=1,6
      180 PDATA(I) = PDA(I,J)
    C**GO TO FORCE PROGRAM
    CALL FORCE (NEXTG,IG4S)
    IF (ERROR .NE. 0) GO TO 460
C
C SAVE FORCE DATA ON UNIT 9 IF REQUIRED
  IF (ISAVE .EQ. 2) GO TO 190
  CALL WRITMS (9,F,11,NEXT)
  WRITE (9,NEXT) F
  NEXT = NEXT + 1
  CALL WRITMS (9,E7,11,NEXT)
  WRITE (9,NEXT) E7
  NEXT = NEXT + 1
190 CONTINUE
C**END OF ALPHA=BETA LOOP INTO FORCE
C*****
C
C
C RESET COUNTERS IN GEOMETRY TABLE IF DATA WERE SAVED
  IF (ISAS .EQ. 0) GO TO 192
  E(1) = NEXTG
  IG4 = 2

```

PRES

PRES 281C
 PRES 282I
 PRES 283
 PRES 284
 PRES 285
 PRES 286
 PRES 287
 PRES 28C
 PRES 289
 PRES 290
 PRES 291
 PRES 292
 PRES 293
 PRES 294
 PRES 295
 PRES 296
 PRES 297
 PRES 298
 PRES 299
 PRES 300
 PRES 301
 PRES 302
 PRES 303
 PRES 304
 PRES 305
 PRES 306
 PRES 307
 PRES 308
 PRES 309
 PRES 310
 PRES 311
 PRES 312
 PRES 313C
 PRES 314I
 PRES 315

PRES

```

C CALL WRITMS (4,E,25,IG4)
C WRTTF (4*IG4) E
C 192 CONTINUE
C IF (IFSAVE .EQ. 2) GO TO 450
C
C**OUTPUT DATA *****
C WRITE HEADER INFORMATION FOR FORCE OUTPUT PAGE
  CALL HEADER
  WRITE (TAPEOT,200) NCOM
  200 FORMAT (1H0,18MCOMPONENT NUMBER #, I3)
  WRITE (TAPEOT,210) (COM(I),I=1,NCB)
  210 FORMAT (1H0,12HPANEL ID # , 20(A4,2X))
  WRITE (6,230) MACH,V,RFND
  230 FORMAT (1H0,7H MACH=#F8,3,6H VEL=#F9,1,16H FT/SEC RF/FT #E13,5 )
  IF (PSTAG .LT. 0.00001) WRITE (6,240) ALT
  240 FORMAT (1H ,7H ALT =F8,0 )
  IF (PSTAG .GT. 0.00001) WRITE (6,250) PSTAG,TSTAG
  250 FORMAT (1H ,16X7HP STAG=#F7,1,16H ATMDS T STAG=#F7,1,6H DEG F )
  WRITE (TAPEOT,260) GTYPE
  260 FORMAT (1H ,2X,2A4)
  WRITE (6,270) SRFF,SPAN,MAC,XCG,YCG,ZCG
  270 FORMAT (1H0,9H S REF =F9,2,8H SPAN =F8,2,8H MAC =F8,2,1H ,
    1 9H X CG =F9,2,8H Y CG =F8,2,8H Z CG =F8,2 )
  WRITE (TAPEOT,280)
  280 FORMAT (1H0,10HFORCE DATA./1H ,7H ALPHA,4X,3HC 0,7X,3HC L,7X,
    1 3HC A,7X,3HC Y,7X,3HC N,1H ,7H BETA ,4X,3ML/D,7X,3HC M,7X,
    2 4HC LL,6X,4HC LN)
C RETRIEVE OUTPUT DATA FROM UNIT 9 AND PRINT
C 300 I=1,NAB
  IG9 = LCOM(NINT) + I*2
  CALL READMS (9,F,11,IG9)
  READ (9*IG9) F
  C WRITE OUTPUT FORCE DATA
  
```

PRES

```

WRITE (TAPEOT,290) F
FORMAT (1H0,F7.2,5F10.5,/1H ,F7.2,4F10.5)
300 CONTINUE
C**OUTPUT DATA COMPLETED FOR THIS COMPONENT
C
C SAVE COMPONENT INFORMATION IN TABLE OF CONTENTS
  DO 191 I=2,11
    191  E2(I) = IPANL(I=1)
        F2(I) = NTOI
        I69 = LCOM(NTOI)
        CALL WRITMS (9,E2,11,I69)
        WRITE (9#IG9) E2
  I69 = I69 + 1
  CALL WRITMS (9,E4,41,I69)
  WRITE (9#IG9) E4
C
C UPDATE MAIN TABLE OF CONTENTS
  E1(1) = NTOI
  E1(2) = ISUM
  E1(3) = NEXT
  I69 = 1
  CALL WRITMS (9,E1,8,I69)
  WRITE (9#IG9) E1
  DO 310 I=2,41
    310  E5(I) = LCOM(I=1)
        F5(I) = 0.0
        I69 = 2
        CALL WRITMS (9,E5,41,I69)
        WRITE (9#IG9) E5
  I69 = 3
  CALL WRITMS (9,E6,41,I69)
  WRITE (9#IG9) F6
  F6(1) = MACH
  E6(2) = V
  F6(3) = REND

```

PRES 316
PRES 317
PRES 318
PRES 319
PRES 320
PRES 321
PRES 322
PRES 323
PRES 324
PRES 325
PRES 326C
PRES 327I
PRES 328
PRES 329C
PRES 330I
PRES 331
PRES 332
PRES 333
PRES 334
PRES 335
PRES 336
PRES 337C
PRES 338I
PRES 339
PRES 340
PRES 341
PRES 342
PRES 343C
PRES 344I
PRES 345
PRES 346C
PRES 347I
PRES 348
PRES 349
PRES 350

PRES

PRES

PRES 351
PRES 352
PRES 353
PRES 354
PRES 355
PRES 356
PRES 357
PRES 358
PRES 359
PRES 360
PRES 361
PRES 362
PRES 363C
PRES 364I
PRES 365
PRES 366
PRES 367
PRES 368
PRES 369
PRES 370
PRES 371
PRES 372
PRES 373
PRES 374
PRES 375
PRES 376
PRES 377I
PRES 378C
PRES 379

```

E8(4) = ALT
E8(5) = PSTAG
E8(6) = TGTAG
E8(7) = GTYPE(1)
E8(8) = GTYPE(2)
E8(9) = SREF
E8(10) = SPAN
E8(11) = MAC
E8(12) = XCG
E8(13) = YCG
E8(14) = ZCG
IG9 = 4
CALL WRITMS (9,E8,14,IG9)
WRITE (9,IG9) F8
WRITE (TAPEOT,430) NTOY
430 FORMAT (1M0,10X,35HDATA SAVFD ON UNIT 9. SET NUMBER #,14)
C
C CHECK IF LAST COMPONENT HAS BFEN COMPLETED
450 IF (NCOM,LT, NCOMP) GO TO 60
GO TO 480
C
C 460 WRITE (TAPEOT,470) ERROR,J
470 FORMAT (1H ,10M** ERROR #,12,26H ON RETURN FROM FORCE. J= ,11)
C 480 RETURN
480 CONTINUE
END

```

PRES

FORCE

FORC	001
FORC	002
FORC	003
FORC	004
FORC	005
FORC	006
FORC	007
FORC	008
FORC	009
FORC	010
FORC	011
FORC	012
FORC	013
FORC	014
FORC	015
FORC	016
FORC	017
FORC	018
FORC	019
FORC	020
FORC	021
FORC	022
FORC	023
FORC	024
FORC	025
FORC	026
FORC	027
FORC	028
FORC	029
FORC	030
FORC	031
FORC	032
FORC	033
FORC	034
FORC	035

```

SUBROUTINE FORCE (NEXT,IG4S)
C
C*****
C**THIS SUBROUTINE CALCULATES THE PRESSURE COEFFICIENT ON EACH SURFACE**
C**ELEMENT. RESOLVES IT INTO THE REQUIRED DIRECTIONS, AND ADDS UP THE **
C**CONTRIBUTIONS TO FIND THE TOTAL VEHICLE COEFFICIENTS. **
C*****
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,RENO,PFS,TFS,AFS,RHOF5,VIS
COMMON /GDATA/LTOT,J,SYMFCT,IORN,IGTYPE,L
COMMON /REF/SREF,MAC,SPAN,XCG,YCG,ZCG
COMMON /TAPE/TAPFIN,TAPENT,TAPFA,TAPB,TAPEC,TAPEJ,TAPEF,TAPEK,
    TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /ABDATA/NAB,ALPHA(20),BETA(20),FOL(20),CDELTA(20),GI(20),
    RI(20),PI(20)
COMMON /FARRAY/F
COMMON /INTERF/INT,ISHE(20),NSHE(100),I9,ISHFF,INF(20,22),
    DINF(20,6),LTTAB(20)
COMMON /GASO/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
COMMON /MDATA/IMPACT,ISHAD,IPRINT,PDATA,ISAVE,IPIN
COMMON /FSBS/FS,BS
DIMENSION ANGLE(3),FS(8),BS(A)
DIMENSION F(11),PDATA(6),E(25),TITLE(15),DINFL(6),LCIND(3),FSS(8)
DIMENSION EI(25),E2(25),E3(25),INFLOW(S)
INTEGER ERROR,SYMFCT,SYMFCO,CASE
INTEGER TAPEJN,TAPEJN,TAPEJN,TAPEJ,TAPEB,TAPEF,TAPEC,TAPEE,TAPEF,
    TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
REAL MACH,MAC,NX,NY,NZ,MACHSO

DATA RC/,17453292E-1/
RAD(A) = A /.572957795E+02

```

FORCE

FORCE

FORC 036
FORC 037
FORC 038
FORC 039
FORC 040
FORC 041
FORC 042
FORC 043
FORC 044
FORC 045
FORC 046
FORC 047
FORC 048
FORC 049
FORC 050
FORC 051
FORC 052
FORC 053
FORC 054
FORC 055
FORC 056
FORC 057
FORC 058
FORC 059
FORC 060
FORC 061
FORC 062
FORC 063
FORC 064
FORC 065
FORC 066
FORC 067
FORC 068
FORC 069
FORC 070

FORCE

C SET UP STARTING CONSTANTS

IFIRST = 0
CA = 0.0
CY = 0.0
CN = 0.0
CLL = 0.0
CLM = 0.0
HML = 0.0
HMF = 0.0
CLN = 0.0
DELTAS = 0.0
AREAT = 0.0
NPRT = 51
NPKX = 51
IDFRIV = 0
IPRCK = 0
IF (IPRINT .EQ. 2) IPRCK = 1
SYMFCO = 0
ISHFI = 0
CPSTAG = PDATA(1)
GGINF = PDATA(2)
ETAC = PDATA(3)
ENPM = PDATA(4)
IMPACI = PDATA(5)
ISHADI = PDATA(6)

C CALCULATE DIRECTION COSINES OF VELOCITY VECTOR

ALPHAR = RAD(ALPHA(J))
SETAR = RAD(BETA(J))
ROLL = RAD(J)
PHIR = RAD(ROLL)
VXO = -V* $\cos(\text{ALPHAR})$ * $\cos(\text{BFTAR})$
VYO = V* $(\sin(\text{PHIR})$ * $\sin(\text{ALPHAR})$ * $\cos(\text{RETAR})$ + $\cos(\text{PHIR})$ * $\sin(\text{BETAR})$)
VZO = V* $(\cos(\text{PHIR})$ * $\sin(\text{ALPHAR})$ * $\cos(\text{BETAR})$ - $\sin(\text{PHIR})$ * $\sin(\text{BETAR})$)

C

FORCE

```
Q = QI(J)
R = RI(J)
P = PI(J)
LTS = LTOT
IF (INT,NE,0 .AND. INF(J,1).GT.1) LTS = LTOTAB(J)
IF (IPIN .GT. 0) LTS = LTOTAB(J)

C INITIALIZE FREE STREAM DATA APRAY
FS(1) = RHOF8
FS(2) = PFS
FS(3) = TFS
FS(4) = AF8
FS(5) = VIS
FS(6) = MACH
FS(7) = MACH * FS(4)
FS(8) = FS(1)*FS(7)/FS(5)
GPI = GAM + 1.0
GMI = GAM = 1.0
DO 9 I=1,8
9 FSS(I) = FS(I)
ICKF = 0
INL = 0

C*****START OF ELEMENT DO LOOP *****
10 DO 509 LM1,LTS
C CHECK FOR SHIELDING OPTION
IF (ISHE1 .EQ. 1) GO TO 15
C CHECK FOR INTERFERENCE
IF (INT,NE,0 .AND. INF(J,1).GT.1) GO TO 16
IF (IPIN .GT. 0) GO TO 16
C GET GEOMETRY DATA FROM NORMAL STORAGE ARRAYS
IG4 = IG4S(L)
CALL READMS (4,E,25,IG4)
READ (4,IG4) E
N = E(2)
```

FORC 071
FORC 072
FORC 073
FORC 074
FORC 075
FORC 076
FORC 077
FORC 078
FORC 079
FORC 080
FORC 081
FORC 082
FORC 083
FORC 084
FORC 085
FORC 086
FORC 087
FORC 088
FORC 089
FORC 090
FORC 091
FORC 092
FORC 093
FORC 094
FORC 095
FORC 096
FORC 097
FORC 098
FORC 099
FORC 100
FORC 101
FORC 102
FORC 103C
FORC 104I
FORC 105

FORCE

FORCE

FORC 106
FORC 107
FORC 108
FORC 109
FORC 110
FORC 111
FORC 112
FORC 113
FORC 114
FORC 115
FORC 116C
FORC 117I
FORC 118
FORC 119C
FORC 120I
FORC 121
FORC 122
FORC 123
FORC 124
FORC 125
FORC 126
FORC 127
FORC 128
FORC 129
FORC 130
FORC 131
FORC 132
FORC 133
FORC 134
FORC 135
FORC 136
FORC 137
FORC 138
FORC 139
FORC 140

```
      M = E(3)
      NX = E(4)
      NY = E(5)
      NZ = E(6)
      XCFNT = E(7)
      YCFNT = E(8)
      ZCFNT = E(9)
      AREA = E(10)
C   OBTAIN POINTER ARRAYS FOR SURFACE DATA SAVE
      IG41 = IG4 + 1
      CALL READMS (4,E2,25,IG41)
      READ (4,IG41) E2
      IG42 = IG4 + 2
      CALL READMS (4,E3,25,IG42)
      READ (4,IG42) E3
C   CHECK ON SYMMETRY REQUIREMENTS (CHANGE SIGNS IF REQUIRED)
      IF (SYMFCO .EQ. 0) GO TO 60
      NY = -NY
      YCENT = -YCENT
      INI = 1
      GO TO 60
C   READ GEOMETRY DATA FROM SHIELDING DATA UNIT
15  READ (TAPEA) E
      GO TO 18
C   READ GEOMETRY DATA FROM FLOW FIELD UNIT
16  READ (TAPEF) IG4,LCOND,E,DINFL,INFLOW
      IG41 = IABS(IG4) + 1
      IF (LCOND(1) .EQ. 2) GO TO 16
C   DUMP GEOMETRY DATA INTO WORKING VARIABLES
1A  N = E(2) + 0.00001
      M = E(3) + 0.0001
      NX = E(4)
      NY = E(5)
      NZ = E(6)
      XCFNT = E(7)
```

FORCE

FORCE

```

      YCENT = E(6)
      ZCENT = E(9)
      AREA = E(10)
      GO CONTINUE
      IF (IPIN .GT. 0) GO TO 68

C   DETERMINE LOCAL EXTERNAL FLOW CONDITIONS
      IF (TNT .NE. 0) GO TO 61
      IF (L .GT. 1) GO TO 69
      GO TO 68

C   INCLUDE INTERFERENCE FLOW FIELD EFFECTS
C   CHECK IF FLOW FIELD IS NON-UNIFORM
      61 IF (DINF(J,1) .GT. 1) GO TO 62
      IF (L .GT. 1) GO TO 69
      DO 65 I=1,9
      65 DINF(I) = DINF(J,I)

C   FLOW FIELD IS NON-UNIFORM, LOCAL FLOW FIELD DATA FROM SAVE UNIT (10)
C   WILL BE USED.
C   CHECK IF LOCAL CONDITIONS ARE NOT FREE-STREAM
      62 IF (LCOND(1) .NE. 0) GO TO 64
      IF (ICKF .FG. 1) GO TO 68
      ICKF = 1

C   SET LOCAL CONDITIONS TO FREESTREAM VALUES
      DO 63 I=1,8
      63 FS(I) = FSS(I)
      GO TO 68

      64 ICKF = 0

C   USE LOCAL FLOW FIELD DATA FROM STORAGE UNIT TO
C   SET UP LOCAL FREE STREAM (FS) TABLE
      FS(2) = DINF(5) * PFS
      FS(3) = DINF(6) * TFS
      FS(1) = GAM*FS(2) / ((GAM-1.0)*GASC*P*FS(3))
      FS(4) = SQRT((GAM-1.0)*GASC*P*FS(3))
      IF (FS(3) .GF. AV1) FS(5) = 2.27E-8*FS(3)**1.5/(FS(3)+198.6)

```

FORC 141
FORC 142
FORC 143
FORC 144
FORC 145
FORC 146
FORC 147
FORC 148
FORC 149
FORC 149
FORC 150
FORC 151
FORC 152
FORC 153
FORC 154
FORC 155
FORC 156
FORC 157
FORC 158
FORC 159
FORC 160
FORC 161
FORC 162
FORC 163
FORC 164
FORC 165
FORC 166
FORC 167
FORC 168
FORC 169
FORC 170
FORC 171
FORC 172
FORC 173
FORC 174
FORC 175

FORCE

FORCE

FORC 176
 FORC 177
 FORC 178
 FORC 179
 FORC 180
 FORC 181
 FORC 182
 FORC 183
 FORC 184
 FORC 185
 FORC 186
 FORC 187
 FORC 188
 FORC 189
 FORC 190
 FORC 191
 FORC 192
 FORC 193
 FORC 194
 FORC 195
 FORC 196
 FORC 197
 FORC 198
 FORC 199
 FORC 200
 FORC 201
 FORC 202
 FORC 203
 FORC 204
 FORC 205
 FORC 206
 FORC 207
 FORC 208
 FORC 209
 FORC 210

FORCE

```

IF (FS(3) .LT. AV1) FS(5) = AV2 * FS(3)**AV3
FS(6) = DINFL(1)
FS(7) = FS(6)*FS(4)
FS(8) = FS(1) * FS(7)/FS(5)
VXJ = FS(7)*DINFL(2)
VYI = FS(7)*DINFL(3)
VZI = FS(7)*DINFL(4)
GO TO 71

C
68 VXI = VXD
VYI = VYG
VZI = VZD
71 VX = VXI
VY = VYI
VZ = VZI

C
69 CONTINUE

C
IF (Q.EQ.0.0 .AND. R.EQ.0.0 .AND. P.EQ.0.0) GO TO 70
CALCULATE VELOCITY COMPONENTS WITH VEHICLE ROTATION
VX = VXI + (Q*(+ZCENT-ZCG)-R*(+YCENT-YCG))
VY = VYI + (R*(+XCENT-XCG)+P*(+ZCENT-ZCG))
VZ = VZI - (P*(+YCENT-YCG)+Q*(+XCENT-XCG))

C
70 VL0CAL = SORT(VX*VX + VY*VY + VZ*VZ)
C COMPUTE COSINE OF ANGLE BETWEEN UNIT VECTORS
COSDEL = (-NX*VX + NY*VY + NZ*VZ) / VL0CAL
IF (COSDEL.GT. 1.0001 .AND. COSDEL.LT. -1.0) COSDEL = -1.0
IF (COSDEL.GT. 1.0 .AND. COSDEL.LT. 1.0001) COSDEL = 1.0
IF (COSDEL.GF. 1.0 .AND. COSDEL.LE. 1.0) GO TO 90
WRITE (6,60)
80 FORMAT (1H0,44H**** FORCE ROUTINE WILL ATTEMPT TO FIND THE
162H ARCCOSINE OF AN ARGUMENT WHOSE ABSOLUTE VALUE IS GREATER THAN
210H ONE *****/1H ,15X30H**** JOB WILL BE TERMINATED **** )
ERROR = 1

```

FORCE

FORC 211
 FORC 212
 FORC 213
 FORC 214
 FORC 215
 FORC 216
 FORC 217
 FORC 218
 FORC 219
 FORC 220
 FORC 221
 FORC 222
 FORC 223
 FORC 224
 FORC 225
 FORC 226
 FORC 227
 FORC 228
 FORC 229
 FORC 230
 FORC 231
 FORC 232
 FORC 233
 FORC 234
 FORC 235
 FORC 236
 FORC 237
 FORC 238
 FORC 239
 FORC 240
 FORC 241
 FORC 242
 FORC 243
 FORC 244
 FORC 245

```

GO TO 620
C COMPUTE ANGLE BETWEEN UNIT VECTORS
C 90 THETA = APCOS(COSDEL)
C
C COMPUTE NEWTONIAN IMPACT ANGLE
C IF (IMPACT.NE.10 .AND. ENPM.NE.0.0) THETA = THETA / FNPM
C DELTA = 1.570796327 * THETA
C IF (DELTA.GT.=0.00001 .AND. DELTA.LT.0.00001) DELTA = 0.0
C
C CALCULATE NEWTONIAN IMPACT ANGLE IN DEGREES
C DELTA = DELTA * .572957795E+02
C ANGLE(2) = DELTA
C
DO 111 I=1,5
111 BS(I) = 0.0
TX = NY*VZ - NZ*VY
TY = NZ*VX - NX*VZ
TZ = NX*VY - NY*VX
SX = TY*NZ - TZ*NY
SY = TZ*NX - TX*NZ
SZ = TX*NY - TY*NX
STOTAL = SQRT(SX*SX + SY*SY + SZ*SZ)
IF (STOTAL.EQ. 0.0) GO TO 115
SX = SX / STOTAL
SY = SY / STOTAL
SZ = SZ / STOTAL
115 CONTINUE
C CHECK IF SURFACE DATA TO BE USED
C IF (IPIN = 1) 119, 116, 117
C CP IS AVAILABLE AS DINF(5)

```

FORCE

FORCE

```

116 CP = DINFL(5)
    CPI = CP
    RS(2) = (0.5*GAM*CP*MACH**2 + 1.0)*PFS
    RS(3) = 1.0
    RS(6) = MACH
    GO TO 415
C
C SURFACE PROPERTIES AVAILABLE IN DINFL ARRAY
117 CP = (DINFL(5) - 1.0)*2.0/(GAM*MACH**2)
    CPI = CP
    BS(2) = DINFL(5)*PFS
    RS(3) = DINFL(6)*IFS
    RS(6) = DINFL(1)
    GO TO 415
119 CONTINUE
C
C CHECK TO SEE IF SURFACE IS IN A SHADOW REGION
    IF (DELTA .LT. 0.0) GO TO 320
C
C ***** SELECT IMPACT PRESSURE METHOD *****
C ***** GO TO (120,130,140,150,160,170,180,190,200,210,240,250,280,290,300
    1      ),IMPACT
C
C CALCULATE PRESSURE USING MODIFIED NEWTONIAN THEORY
120 CP = CPSTAG * COSDEL * COSDEL
    GO TO 410
C
C CALCULATE PRESSURE USING NEWTONIAN - PRANDTL-MEYER METHOD
130 ISE = 1
    CALL NEWTPM (ANGLE,EMN,CP,ETAC,IPRCK,MFR,CPSTAG,
    1      ISE,IFIRST)

```

FORC 246
FORC 247
FORC 248
FORC 249
FORC 250
FORC 251
FORC 252
FORC 253
FORC 254
FORC 255
FORC 256
FORC 257
FORC 258
FORC 259
FORC 260
FORC 261
FORC 262
FORC 263
FORC 264
FORC 265
FORC 266
FORC 267
FORC 268
FORC 269
FORC 270
FORC 271
FORC 272
FORC 273
FORC 274
FORC 275
FORC 276
FORC 277
FORC 278
FORC 279
FORC 280

FORCE

FORCE

```

      GO TO 410
C   C   CALCULATE PRESSURE USING TANGENT WEDGE
      ISDFY = 2
      CALL COMPR (ANGLE, MER, IPRCK, CPSTAG, ISDFT, JFIRST, CP)
      GO TO 410
C   C   CALCULATE PRESSURE USING TANGENT WEDGE = INFINITE MACH METHOD
      EMNS = 0.5*(GAM+1.)*FS(6)*SIN(DELTA)+FXP(-0.25*(GAM+1.))*
      FS(6)*SIN(DELTA))
      CP = 4./(GAM+1.)*(EMNS**2-1.)/(FS(6)*FS(6))
      GO TO 410
C   C   CALCULATE PRESSURE USING TANGENT CUNE EMPIRICAL METHOD
      ANGLE(1) = DELTA
      CALL CONF (ANGLE, CP, 2)
      GO TO 410
C   C   CALCULATE PRESSURE USING CONE AT ANGLE OF ATTACK
      CONTINUE
C   C   JONES METHOD IS USED
      IT = 0
C   C   CALCULATE LOCATION OF WINDWARD PLANE
      PHIW = 0.0
      IF (VY .EQ. 0.0 .AND. VZ .GE. 0.0) GO TO 176
      PHIW = 3.1415926536
      IF (VY .EQ. 0.0) GO TO 176
      PHIW = ATAN2(-VY, VZ)
      CONTINUE
C   C   ANGLE OF ATTACK IN WINDWARD PLANE
      ALFWD = ARCOS(-VX/FS(7))
C   C   CONE ANGLE
      DCR = ARSIN(NX)
FORC 281
FORC 282
FORC 283
FORC 284
FORC 285
FORC 286
FORC 287
FORC 288
FORC 289
FORC 290
FORC 291
FORC 292
FORC 293
FORC 294
FORC 295
FORC 296
FORC 297
FORC 298
FORC 299
FORC 300
FORC 301
FORC 302
FORC 303
FORC 304
FORC 305
FORC 306
FORC 307
FORC 308
FORC 309
FORC 310
FORC 311
FORC 312
FORC 313
FORC 314
FORC 315
```

FORCE

FORCE

```

C C MFRIDIAN ANGLE
  IF (NY,NE, 0.0) GO TO 177
  PHI = 0.0
  IF (NZ,EG, 0.0) GO TO 17A
  177 PHI = ATAN2(NY, -NZ)

C C LOCATION FROM HINDWARD PLANE
  17A PHIT = PHI - PHIW

C C MACH NUMBER IN PHIT PLANE
  VC = VZ*COS(PHI) - VY*SIN(PHI)
  AMP = FS(6)*SQRT(VX**2 + VC**2)/FS(7)

C C ANGLE OF AMP TO AXIS
  IF (VC,NE, 0.0) GO TO 179
  ALFP = 0.0
  IF (VX,EG, 0.0) GO TO 181
  179 ALFP = ATAN2( VC, -VX)

C C
  181 ANGLE(1) = DCR/RC
  IF (IPRCK,EG, 1)
    WRITE(6,100) PHIW, PHI,PHIT, ALFWD,ALFP,DCR,AMP
  1000 FORMAT(1H0, 7X0PHIW,11X3PHI, 12X4MPHTT, 11X5HALFWD,
    1 10X4HALFP, 11X3HDCR, 12X3HAMP/1H , 7F15.6)
    CALL ACONE(ANGLE,CP,ALFWD,PHIT,ALFP,AMP,IT,IPRCK)

C C GO TO 410

C C CALCULATE PRESSURE USING VAN DYKE UNIFIED THEORY
  180 CP = 0.5*(GAM+1.)*DELTA**2 + SORT((0.25*(GAM+1.))**2*DELTA**4
    1 + 4.0*DELTA**2/((FS(6)*FS(6))-1.0))
    GO TO 41C

C

```

FORC 316
 FORC 317
 FORC 318
 FORC 319
 FORC 320
 FORC 321
 FORC 322
 FORC 323
 FORC 324
 FORC 325
 FORC 326
 FORC 327
 FORC 328
 FORC 329
 FORC 330
 FORC 331
 FORC 332
 FORC 333
 FORC 334
 FORC 335
 FORC 336
 FORC 337
 FORC 338
 FORC 339
 FORC 340
 FORC 341
 FORC 342
 FORC 343
 FORC 344
 FORC 345
 FORC 346
 FORC 347
 FORC 348
 FORC 349
 FORC 350

FORCE

FORCE

```

C CALCULATE BLUNT BODY VISCOUS EFFECTS
190 CP = 0.0
   IVISIN = 1
C THE VISCOUS FORCE COEFFICIENT TAU IS CALCULATED IN
C SUBROUTINE BLUNT, WHICH ONLY NEEDS TO BE CALLED ONCE
C FOR EACH SECTION.
C
   IF(L.EQ.1)CALL BLUNT(PFS,FS(6),TFS,VIS,RMGFS,ETAC,RENO,TAU,IVISIN)
   SHEAR = TAU*COR(DELTAR)
   GO TO 220
C CALCULATE PRESSURE USING SHOCK-EXPANSION METHOD
200 CALL SHKEXP (N,M,IPRCK,NX,NY,N7,DELTA,PFS,FS(6),
1 DELTAR,IMPACT,CPSTAG,TFS,CP,ISHADI,IFIRST,L,IPRINT,
2 RHOFS,AFS,VIS,V,RENO,0)
   IF (ERROR .EQ. 3) GO TO 620
   GO TO 410
C CALCULATE PRESSURE USING FREE MOLECULAR FLOW
210 FN = CPSTAG
   FT = ENPM
   TBTIN = ETAC
   S = SORT(GAM/2.0)*FS(6)
   SSIND = S * SIN(DELTAR)
   ERFS = ERF(SSIND)
   CP = (1.0/(S*S))*(((2.0*FN)/(1.7724539*SSIND*FN/2.0*SQRT(TBTIN)))*)
1 EXP(-SSIND*SSIND)) + (((2.0*FN)/(S*SSIND*SSIND*0.5)*FN/2.0*
2 1.7724539*SQRT(TBTIN)*SSIND) * (1.0*ERFS) ))
C CALCULATE SHEAR FORCE COEFFICIENT FOR FREE MOLECULAR FLOW
   SHFAR = COS(DELTAR)*FT/(1.7724539*S) * (EXP(-SSIND*SSIND)
1 + 1.7724539*SSIND*(1.0*ERFS))
   SKIN = SHEAR
220 IF (SHFAR .GT. 1.0E-25) GO TO 230
   SHEARX = 0.0

```

FORCE

FORCE

FORC 386
 FORC 387
 FORC 388
 FORC 389
 FORC 390
 FORC 391
 FORC 392
 FORC 393
 FORC 394
 FORC 395
 FORC 396
 FORC 397
 FORC 398
 FORC 399
 FORC 400
 FORC 401
 FORC 402
 FORC 403
 FORC 404
 FORC 405
 FORC 406
 FORC 407
 FORC 408
 FORC 409
 FORC 410
 FORC 411
 FORC 412
 FORC 413
 FORC 414
 FORC 415
 FORC 416
 FORC 417
 FORC 418
 FORC 419
 FORC 420

```

    SHFARY = 0.0
    SHEARZ = 0.0
    GO TO 410
230 SHFARX = SHEAR * SX
    SHFARY = SHEAR * SY
    SHFARZ = SHEAR * SZ
    GO TO 410
C
C SET PRESSURE COEFFICIENT TO INPUT VALUE
240 CP = CPSTAG
    GO TO 410
C
C CALCULATE PRESSURE USING HANKFY FLAT SURFACE EMPIRICAL CORRELATION
250 IF (DELTA .GE. 10.0) GO TO 260
    HANKFY = (0.195+0.222594/FS(6))*0.3-0.4)*DELTA + 4.0
    GO TO 270
260 HANKFY = 1.95 + 0.3925/(FS(6))*0.3*TAN(DELTA))
270 CP = HANKFY * COSDEL * COSDFL
    GO TO 410
C
C CALCULATE PRESSURE USING DELTA WING CORRELATION (SMYTH)
280 DELDLW = DELTAR
    IF (DELDLW .LT. 0.01745) DELDLW = 0.01745
    EMNS = FS(6) * SIN(DELDLW)
    EMNS = 1.09 * EMNS + EXP(-0.49 * EMNS)
    CP = 1.0 / (FS(6) * FS(6)) * 1.66667 * (EMNS * EMNS + 1.0)
    GO TO 410
C
C CALCULATE PRESSURE USING MODIFIED DALHEM-BULK RELATIONSHIP
290 XLNM = ALOG(FS(6))
    A1 = (6.7 * 3 * FS(6)) * SIN((XLNM - 5.88) / 1.20 * 3.14159)
    A2 = 1.15 + 0.5 * SIN((XLNM - 0.916) / 3.2 * 3.14159)
    IF (OFLTAR .EQ. 0.0) CP = 0.0
    IF (DFLTAR .EQ. 0.0) GO TO 410
  
```

FORCE

FORCE

```

CP=1.0/(ABS(SIN(4.0*DELTA)))*0.75+1.0
IF(CP.GT.5.0) CP=5.0
IF(CP.LT.2.0) OR. (DELTA.GT.22.5)) CP=2.0
CP=CP*COSEDEL*COSEDEL*(AI/DELTA**A2+1)
GO TO 410

C
C CALCULATE PRESSURE USING BLAST WAVE ANALYSIS
300 IF (CPSTAG.GT. 0.5) GO TO 310
CP = (0.067*(FS(6)*FS(6))*ETAC/(ENPM-XICENT) + 0.44 )
1 / (GAM/2.0*(FS(6)*FS(6)))
GO TO 410
310 CP = (0.121*(FS(6)*FS(6))*ETAC/(ENPM-XICENT)**.667 + 0.56)
1 / (GAM/2.0*(FS(6)*FS(6)))
GO TO 410

C
C*****
C***** SELECT SHADOW PRESSURE METHOD *****
C*****
320 GO TO (330,340,350,170,370,380,200,390,210),TSHAD

C
C CALCULATE PRESSURE IN SHADOW REGIONS
C
C CALCULATE PRESSURE USING CP=0 IN SHADOW REGIONS
330 CP = 0.0
GO TO 410

C
C CALCULATE PRESSURE USING NEWTONIAN = PRANDTL-MEYER
340 ISE = 1
CALL NEWTPM (ANGLE,EMN,CP,ETAC,IPRCK,MFR,CPSTAG,
1 ISE,IFIRST )
GO TO 410

C
C CALCULATE PRESSURE USING PRANDTL-MEYER EXPANSION FROM FREE STREAM
350 ANGLE(2) = ABS(DELTA)
ISEDET = 2

```

FORC 421
FORC 422
FORC 423
FORC 424
FORC 425
FORC 426
FORC 427
FORC 428
FORC 429
FORC 430
FORC 431
FORC 432
FORC 433
FORC 434
FORC 435
FORC 436
FORC 437
FORC 438
FORC 439
FORC 440
FORC 441
FORC 442
FORC 443
FORC 444
FORC 445
FORC 446
FORC 447
FORC 448
FORC 449
FORC 450
FORC 451
FORC 452
FORC 453
FORC 454
FORC 455

FORCE

FORCE

FORC 456
 FORC 457
 FORC 458
 FORC 459
 FORC 460
 FORC 461
 FORC 462
 FORC 463
 FORC 464
 FORC 465
 FORC 466
 FORC 467
 FORC 468
 FORC 469
 FORC 470
 FORC 471
 FORC 472
 FORC 473
 FORC 474
 FORC 475
 FORC 476
 FORC 477
 FORC 478
 FORC 479
 FORC 480
 FORC 481
 FORC 482
 FORC 483
 FORC 484
 FORC 485
 FORC 486
 FORC 487
 FORC 488
 FORC 489
 FORC 490

```

C      CALL EXPAND (ANGLE, MER, IPRCK, ISDET, CP)
C      GO TO 410
C
C      CALCULATE PRESSURE USING VAN DYKE UNIFIED THEORY
370 CP = -1.4/(GAM*FS(6)**2)
    B = SQRT(FS(6)**2 - 1.0)
    DELMX = 2.*(1. - (1. - 0.7*(B/FS(6))**2)**((GAM-1.)*0.5/GAM))
    1 / (B*(GAM-1.0))
    IF (ABS(DELTA) .GE. DELMX) GO TO 410
    CP = 2./((GAM*B**2)**((1.-.5*(GAM-1.))*B*ABS(DELTA)))
    1 **((2.*GAM/(GAM-1.)) - 1.0)
    GO TO 410
C
C      CALCULATE PRESSURE USING RASE PRESSURE RELATIONSHIP (CP = 1/M**2)
380 CP = - 1.0 / (FS(6)*FS(6))
    GO TO 410
C
C      SET PRESSURE COEFFICIENT TO INPUT VALUE
390 CP = ETAC
    GO TO 410
C
C *****
410 CONTINUE
C *****
C
C      CORRECT CP FOR LOCAL Q
    CP = CP * GOINF * VLOCAL*VLOCAL / (V*V)
C      CORRECT CP TO FREE STREAM CONDITIONS IF INTERFERENCE WAS USED
    IF (INT .EQ. 0) GO TO 415
    IF (LCOND(1) .EQ. 0) GO TO 415
    CPI = CP
    CP = ((CP*GAM/2.*(FS(6)*FS(6))**1.)* OINFL(5)-1.0) /

```

FORCE

FORCE

```
      1 (GAM/2, *MACH*MACH)
415 CONTINUE
C
C CHECK IF LOCAL FLOW PROPERTIES (BS ARRAY) ARE NEEDED,
  IF (ISAVE.EQ. 0) GO TO 416
C
  IF (DELTA .GE. 0)
150 TO (417,418,416,417,416,417,416,417,416,416,416,
2    417,417,417,417), IMPACT
C
  GO TO (417,418,416,417,417,417,416,417,417), ISHAD
C
418 IF (MER.EQ. 0) GO TO 416
C
C CALCULATE APPROXIMATE LOCAL CONDITIONS, RS; GIVEN ONLY CP
417 CONTINUE
  IF (ABS(CP) .LT. 0.00001) GO TO 402
  IF (CP) 403,402,401
C
  CP = 0.0, SET RS = FS
402 DO 391 I = 1,8
391  RS(I) = FS(I)
  GO TO 416
C
C APPROXIMATE LOCAL FLOW ASSUMING EXPANSION FROM FS.
C PRESSURE (FROM DEFINITION OF CP)
403 MACHSQ = MACH**2
  P2P1 = 0.5*(GAM*MACHSQ*CP + 1.0
  BS(2) = PFS*P2P1
  P2P1 = BS(2)/FS(2)
C TEMPERATURE
  T2T1 = (P2P1)**(GAM/(GAM-1))
  BS(3) = FS(3)*T2T1
C SPEED OF SOUND
  BS(4) = FS(4)*SORT(T2T1)
```

FORCE

FORCE

FORC 526
 FORC 527
 FORC 528
 FOPC 529
 FORC 530
 FORC 531
 FORC 532
 FORC 533
 FORC 534
 FORC 535
 FORC 536
 FORC 537
 FORC 538
 FORC 539
 FORC 540
 FORC 541
 FORC 542
 FORC 543
 FORC 544
 FORC 545
 FORC 546
 FORC 547
 FORC 548
 FORC 549
 FORC 550
 FORC 551
 FORC 552
 FORC 553
 FORC 554
 FORC 555
 FORC 556
 FOPC 557
 FORC 558
 FORC 559
 FORC 560

```

C DENSITY
  RS(1) = FS(1)*P2P1/T2T1
C MACH NUMBER
  BS(6) = SQRT(2.0/GM1*((1.0+0.5*GM1*FS(4)**2)/T2T1 - 1.0))
C VELOCITY
  RS(7) = BS(6)*RS(4)
C VISCOSITY
  IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.)
  IF(RS(3).LT.AV1) BS(5) = AV2*BS(3)**AV3
C REYNOLDS NUMBER
  BS(8) = BS(1)*RS(7)/RS(5)
C
  GO TO 416
C
C CALCULATE LOCAL CONDITIONS, BS, GIVEN ONLY CP.
C APPROXIMATED ASSUMING ORLIQUE SHOCK WAVE.
C
C PRESSURE (FROM DEFINITION OF CP)
#01 MACHSQ = MACH*MACH
  P2P1 = 0.5*GM1*MACHSQ*CP + 1.0
  BS(2) = PFS*P2P1
  P2P1 = BS(2)/FS(2)
C DENSITY (EQ. 158)
  R2P1 = (GP1*P2P1 + GM1)/(GM1*P2P1 + GP1)
  RS(1) = FS(1)*R2P1
C TEMPERATURE (EQ. 159)
  T2T1 = P2P1/P2P1
  RS(3) = FS(3)*T2T1
C SPEED OF SOUND
  BS(4) = FS(4)*SQRT(T2T1)
C
C VISCOSITY
  IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.6)
  IF(BS(3).LT.AV1) BS(5) = AV2*BS(3)**AV3

```

FORCE

FORCE

```

C MACH NUMBER (EQ, 157)
  BS(6) = SQRT((MACHSQ*(GP1*P2P1 + GM1) - 2.0*(P2P1**2 - 1.0))
  1 / (P2P1*(GM1*P2P1 + GP1)))
  IF (BS(6) .LT. 1.0) BS(6) = 1.01
C VELOCITY
  BS(7) = BS(4) * BS(6)
C REYNOLDS NUMBER PER FOOT
  BS(8) = BS(1) * BS(7)/RS(5)
C
C 416 CONTINUE
C
C*****
C***** PRESSURE CALCULATION PART OF PROGRAM HAS BEEN COMPLETED *****
C*****
C
C
C
C SELECT FORCE METHOD TO MEET SYMMETRY REQUIREMENTS
  IF (SYMFC, EQ, 1) GO TO 420
  IF (P, NE, 0.0) .OR. P, NE, 0.0) GO TO 420
  IF (RETA(J), EQ, 0.0) .AND. ROLL, EQ, 0.0) GO TO 450
C
C CALCULATE SIX-COMPONENT FORCE CONTRIBUTIONS OF ELEMENT
420 DELCA = NX * (CP*AREA/SREF)
  DELCY = NY * (CP*AREA/SREF)
  DELCN = NZ * (CP*AREA/SREF)
  IF (IMPACT, EQ, 8) .AND. DELTAR, GE, 0.0) GO TO 430
  IF ((IMPACT, EQ, 10) .AND. DELTAR, GE, 0.0) .OR.
  1 (ISHAD, EQ, 9) .AND. DELTAR, LT, 0.0) GO TO 430
  GO TO 440
430 DELCA = DELCA + SHEARX *(AREA/SREF)
  DELCY = DELCY + SHEARY *(AREA/SREF)
  DELCN = DELCN + SHEARZ *(AREA/SHEFF)
440 DELCLL = DELCY * (ZCENT - ZCG)/SPAN

```

FORC 561
FORC 562
FORC 563
FORC 564
FORC 565
FORC 566
FORC 567
FORC 568
FORC 569
FORC 570
FORC 571
FORC 572
FORC 573
FORC 574
FORC 575
FORC 576
FORC 577
FORC 578
FORC 579
FORC 580
FORC 581
FORC 582
FORC 583
FORC 584
FORC 585
FORC 586
FORC 587
FORC 588
FORC 589
FORC 590
FORC 591
FORC 592
FORC 593
FORC 594
FORC 595

FORCE

FORCE

596 FORC
 597 FORC
 598 FORC
 599 FORC
 600 FORC
 601 FORC
 602 FORC
 603 FORC
 604 FORC
 605 FORC
 606 FORC
 607 FORC
 608 FORC
 609 FORC
 610 FORC
 611 FORC
 612 FORC
 613 FORC
 614 FORC
 615 FORC
 616 FORC
 617 FORC
 618 FORC
 619 FORC
 620 FORC
 621 FORC
 622 FORC
 623 FORC
 624 FORC
 625 FORC
 626 FORC
 627 FORC
 628 FORC
 629 FORC
 630 FORC

```

1  +DELCLN * (YCENT - YCG)/SPAN
1  DELCLM= DELCN * (XCENT - XCG)/MAC
1  +DELCA * (ZCENT - ZCG)/MAC
1  DELCLN= DELCY * (XCENT - XCG)/SPAN
1  -DELCA * (YCENT - YCG)/SPAN
GO TO 460
450 DELCA = NX * (CP*AREA/SREF) * 2.0
DELCY = 0.0
DELCN = NZ * (CP*AREA/SREF) * 2.0
IF (IMPACT.EQ.17 .OR. ISHAD.EQ.10) GO TO 470
IF (IMPACT.EQ.8 .AND. DELTAR.GE.0.0) GO TO 460
IF ((IMPACT.EQ.10.AND.DELTAR.GE.0.0) .OR.
1  (ISHAD.EQ.9.AND.DELTAR.LT.0.0)) GO TO 460
GO TO 470
460 DELCA = DELCA + SHEARX * 2.0*(AREA/SREF)
DELCN = DELCN + SHEARZ * 2.0*(AREA/SREF)
470 DELCLL = 0.0
DELCM=(DELCN * (XCENT + XCG) / MAC
1  +DELCA * (ZCENT - ZCG) / MAC )
DELCN= 0.0
C SUM UP SIX-COMPONENT FORCE CONTRIBUIONS
480 CA = CA + DELCA
CY = CY + DELCY
CN = CN + DELCN
CLL = CLL + DELCLL
CLM = CLM + DELCLM
CLN = CLN + DELCLN
APFAT = AREAT + AREA
490 IF (UPRINT .EQ. 0) GO TO 540
CHECK TO PRINT HEADER AT TOP OF PAGE
500 IF (NPRT.LT.NPCK) GO TO 560
510 NPRT = 0
CALL HEADER
WRITE (TAPEUT,520) (GTYPE(I),I=1,2)
  
```

FORCE

FORCE

FGRC 631
 FORC 632
 FORC 633
 FORC 634
 FORC 635
 FORC 636
 FORC 637
 FORC 638
 FORC 639
 FORC 640
 FORC 641
 FORC 642
 FORC 643
 FORC 644
 FORC 645
 FORC 646
 FORC 647
 FORC 648
 FORC 649
 FORC 650
 FORC 651
 FORC 652
 FORC 653
 FORC 654
 FORC 655
 FORC 656
 FORC 657
 FORC 658
 FORC 659
 FORC 660
 FORC 661
 FORC 662
 FORC 663
 FORC 664
 FORC 665

```

520 FORMAT (1H , 63X, 2A4)
530 CONTINUE
WRITE (TAPEOT,540) MACH,ALT,SREF,SPAN,IMPACT,IMPACTI,
1 XCG,YCG,ZCG,MAC,ISHAD,ISHADI
540 FORMAT (1H0,20ELEMENT DATA MACH#F7.3,7M ALT #F8.0,9M S REF #
1 F8.1,8M SPAN #F7.1,10M IMPACT #I3,10M IMPACTI #I3,/1M ,
2 15X5HXCG #F7.1,7H YCG #F7.1,10H ZCG #F7.1,4X5HMAC #F7.1,
3 10H ISHAD #I3,10H ISHADI #I3)
WRITE (TAPEOT,550) ALPHA(J),BETA(J),CPSTAG,ETAC,DELTAS,
1 IDERIV,Q,R,P
550 FORMAT (1H ,5X)7TH ANGLE OF ATTACK #F6.2,3X)11HYAW ANGLE #F6.2,
1 3X)3HK # F8.5,3X)6METAC #F8.4,3X,9H)DELTA E #F6.2,/1M ,
2 5X)8)HIDERIV #I3,3X)3HQ #E12.5,3X,3HR #F12.5,3X)3IMP #E12.5,/
3 1H0,2X,1H),6X,6H)DEL CA,8X
4 6H)DEL CY,6X)6H)DEL CN,7X)7H)DEL CLL,7X)7H)DEL CLM,7X)7H)DEL CLN,7X)2H)CP,
5 13X)4H)AREA,/1H ,11X)2H)CA,12X)2H)CY,12X)2H)CN,11X)3H)CLL,11X)3H)CLM,
6 12X)3H)CLN,8X)5H)DELTA/1H ,10X)5H)XCENT, 9X)5H)ZCENT, 9X)5H)ZCENT)
NPRT # NPRT + 15

C PRINT ELEMENT DATA
560 WRITE (TAPEOT,570) L,DELCA,DELCO,DELCOV,DELCCL,DELCCLM,DELCCLN,CP,
1 AREA,CA,CY,CN,CLL,CLM,CLN,DELTA,XCENT,YCENT,ZCENT
570 FORMAT (1H0,14,8E14.5,/1H ,4X)7E14.5/1M , 4X)3E14.5)
NPRT # NPRT + 4
IF (INT .EQ. 0) GO TO 580
IF (LCOND(1) .EQ. 0) GO TO 580
WRITE (TAPEOT,585) LCOND,CPI,DINFL,INFLOW
NPRT # NPRT + 2
585 FORMAT (1H ,8X,6H)LCOND#I2,I4,I3,6M CPI#E13.5,8M DINFL#E69.5/
1 1M , 6X)16H)FLOW DATA SET # , I2, 4X)17H)ALPHA#BETA SET # , I2,
2 4X)14H)FLOW REGION # , I2, 4X)13H)SUB-REGION # , I2,
3 4X)17H)SECONDARY FLOW # , I2)

C SET UP AND SAVE ELEMENT RESULTS ON GEOMETRY STORAGE UNIT.
C ONLY DATA FOR ELEMENTS ACTUALLY ON UNIT 4 ARE SAVED.

```

FORCE

FORCE

FORC	666
FORC	667
FORC	668
FORC	669
FORC	670
FORC	671
FORC	672
FORC	673
FORC	674
FORC	675
FORC	676
FORC	677
FORC	678
FORC	679
FORC	680
FORC	681
FORC	682
FORC	683
FORC	684
FORC	685
FORC	686
FORC	687
FORC	688
FORC	689
FORC	690
FORC	691
FORC	692
FORC	693
FORC	694
FORC	695
FORC	696
FORC	697
FORC	698
FORC	699
FORC	700

C DATA FOR REFLECTED ELEMENTS, PARTIAL ELEMENTS, ETC., ARE NOT SAVED.

580 IF (ISAVE .EQ. 0) GO TO 589

IF (Y64 .LE. 0) GO TO 589

E1(1) = IG4

E1(2) = E(1)

E1(3) = 0.0

E1(4) = 0.0

E1(5) = CP

E1(6) = DELTA

E1(7) = RS(6)

E1(8) = SX

E1(9) = SY

E1(10) = SZ

E1(11) = BS(2)/PFS

E1(12) = BS(3)/TFS

E1(13) = DELCA

E1(14) = DELCY

E1(15) = DELCN

E1(16) = DELCLL

E1(17) = DELCLM

E1(18) = DELCLN

E1(19) = 0.0

E1(20) = 0.0

E1(21) = 0.0

E1(22) = 0.0

E1(23) = 0.0

E1(24) = 0.0

E1(25) = 0.0

C CHECK IF SPACE IS AVAILABLE FOR STORING ELEMENT DATA

IF (NEXT .LE. 3000) GO TO 585

WRITE (TAPE07,583)

583 FORMAT (1H0,42HSPACE ON UNIT 4 FOR SURFACE DATA HAS BEEN ,

1 37HEXCEEDED; NO MORE DATA WILL BE SAVED.)

ISAVE = 0

GO TO 589

FORCE

FORCE

FORC 701
FORC 702C
FORC 703I
FORC 704
FORC 705
FORC 706
FORC 707C
FORC 708I
FORC 709
FORC 710
FORC 711C
FORC 712I
FORC 713
FORC 714
FORC 715
FORC 716
FORC 717
FORC 718
FORC 719
FORC 720
FORC 721
FORC 722
FORC 723
FORC 724
FORC 725
FORC 726
FORC 727
FORC 728
FORC 729
FORC 730
FORC 731
FORC 732
FORC 733
FORC 734
FORC 735

```
585 CONTINUE
CALL WRITMS (4,E1,25,NEXT)
WRITE (4#NEXT) E1
C RESET POINTER ARRAY FOR SAVED DATA
IF (INL .EQ. 1) GO TO 587
F2(J) = NEXT
CALL WRITMS (4,E2,25,IG41)
WRITE (4#IG41) E2
GO TO 586
587 E3(J) = NEXT
CALL WRITMS (4,E3,25,IG42)
WRITE (4#IG42) E3
C 588 NEXT = NEXT + 1
IF (IPRINT .EQ. 0) GO TO 589
WRITE (TAPEOT,591) E1,F2
C 591 FORMAT (1H,3HE12.5,7IH,3X,8E12.5,7IH,3X,9E12.5,7IH,
C 1 3HE22.8E12.5,7IH,3X,8E12.5,7IH,3X,9F12.5)
C NPRT = NPRT + 6
589 CONTINUE
C ***** END OF DO LOOP FOR ALL ELEMENTS
C
C CHECK IF SHIELDING WAS DONE AND COMPLETED
IF (ISHE1 .EQ. 1) GO TO 610
C CHECK IF SHIELDING IS TO BE DONE BUT IS NOT COMPLETED
IF (ISHEF .NE. 0) GO TO 600
C CHECK IF NO SYMMETRY
IF (SYMFCO .EQ. 1) GO TO 610
C CHECK IF ROTATION RATES ARE USED
IF (P.NE.0.0 .OR. P.NE.0.0) GO TO 590
C CHECK IF YAW OR ROLL ARE ZERO
IF (BETA(J),FO.0.0 .AND. ROLL.FO.0.0) GO TO 610
C CHECK IF SYMMETRY CALCULATIONS ARE COMPLETED
590 IF (SYMFCO .EQ. 1) GO TO 610
C CHECK IF INTERFERENCE TABLES WERE USED (SYMMETRY ALREADY DONE)
```

FORCE

FORCE

```
IF (INT,NE,0 .AND. INF(J,1).GT.1) GO TO 610
C SET SYMMETRY RECYCLE FLAG
  SYMFCO = 1
  GO TO 10
C
C
  500 I9 = I9 + 1
  LTS = NSHE(I9)
  IF (LTS .EQ. 0) GO TO 610
  ISWF1 = 1
  GO TO 10
C
C
  610 CONTINUE
C SET UP ARRAY OF FINAL DATA
  F(1) = ALPHA(J)
  F(4) = CA
  F(5) = CY
  F(6) = CN
  F(7) = BETA(J)
  F(9) = CLM
  F(10) = CLL
  F(11) = CLN
C RESOLVE NORMAL AND AXIAL FORCES IN LIFT AND DRAG DIRECTION
  CN = CA*COS(ALPHAR)*COS(RETAR)-CY*SIN(PHIR)*SIN(ALPHAR)*COS(BETAR)
  1 -CY*COS(PHIR)*SIN(RETAR)+CN*COS(PHIR)*SIN(ALPHAR)*COS(RETAR)
  2 -CN*SIN(PHIR)*SIN(RETAR)
  CL = -CA*SIN(ALPHAR)-CY*SIN(PHIR)*COS(ALPHAR)+CN*COS(PHIR)
  1 *COS(ALPHAR)
C CALCULATE SIDE FORCE COEFFICIENT - WIND AXIS
  CYPRTM = CA*COS(ALPHAR)*SIN(BETAR)-CY*SIN(PHIR)*SIN(ALPHAR)
  1 *SIN(BETAR)+CY*COS(PHIR)*COS(BETAR)+CN*COS(PHIR)*SIN(ALPHAR)
  2 *SIN(BETAR)+CN*SIN(PHIR)*COS(BETAR)
C SET UP REMAINING PARAMETERS
```

FORC 736
FORC 737
FORC 738
FORC 739
FORC 740
FORC 741
FORC 742
FORC 743
FORC 744
FORC 745
FORC 746
FORC 747
FORC 748
FORC 749
FORC 750
FORC 751
FORC 752
FORC 753
FORC 754
FORC 755
FORC 756
FORC 757
FORC 758
FORC 759
FORC 760
FORC 761
FORC 762
FORC 763
FORC 764
FORC 765
FORC 766
FORC 767
FORC 768
FORC 769
FORC 770

FORCE

FORCE

```
F(2) = CD  
F(3) = CL  
IF (CD .EQ. 0.0) CD = 0.000001  
F(8) = CL / CD
```

```
C  
C 620 RETURN  
END
```

```
FORC 771  
FORC 772  
FORC 773  
FORC 774  
FORC 775  
FORC 776  
FORC 777  
FORC 778
```

FORCE

```

SUBROUTINE SHKEXP(N,M,IPRCK,NX,NY,NZ,DELTA,PFS,MACH,
1 DELTAR,IMPACT,CPSTAG,IFS,CP,ISHADI,IFIRST,LL,IPRINT,
2 RHDFS,AFS,VIS,V,RENO,ISMDF)
C
C*****
C*** THIS SUBROUTINE CALCULATES LOCAL CONDITIONS ON AN ELEMENT USING ***
C*** SHOCK EXPANSION THEORY. ***
C*****
C
COMMON /EXEC/CASE, TITLE, PAGE, EPROR
COMMON /GDATA/LTNT,J,SYMFCT,IORN,IGTYPE,L
COMMON /FSBS/FS,RS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
COMMON /TAPE/TAPFIN,TAPEOT,TAPEA,TAPFS,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
DIMENSION YTITLE(15),ANGLE(3),FS(8),RS(8),NXI(30),NYI(30),
1 NZI(30),PSS(R,30),FSS(8)
INTEGER CASE,PAGE,ERROR
INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
REAL MACH,NX,NY,NZ,NXI,NYI,NZI,NU
C
IOPIEN = IORN
DO 5 I=1,8
5 FSS(I) = FS(I)
C
C CHECK IF THIS ELEMENT IS THE FIRST ELEMENT IN A COLUMN OF ELEMENTS
IF( IORIEN.EQ.0.AND.N.EQ.1 .OR. IORIEN.EQ.1.AND.M.EQ.1 ) GO TO 60
C
C*****
C***** CALCULATIONS FOR ELEMENTS THAT ARE NOT INITIAL ROW ELEMENTS ***
C*****

```

SHKE 001
SHKE 002
SHKE 003
SHKE 004
SHKE 005
SHKE 006
SHKE 007
SHKE 008
SHKE 009
SHKE 010
SHKE 011
SHKE 012
SHKE 013
SHKE 014
SHKE 015
SHKE 016
SHKE 017
SHKE 018
SHKE 019
SHKE 020
SHKE 021
SHKE 022
SHKE 023
SHKE 024
SHKE 025
SHKE 026
SHKE 027
SHKE 028
SHKE 029
SHKE 030
SHKE 031
SHKE 032
SHKE 033
SHKE 034
SHKE 035

SHKEXP

```
SHKE 036
SHKE 037
SHKE 038
SHKE 039
SHKE 040
SHKE 041
SHKE 042
SHKE 043
SHKE 044
SHKE 045
SHKE 046
SHKE 047
SHKE 048
SHKE 049
SHKE 050
SHKE 051
SHKE 052
SHKE 053
SHKE 054
SHKE 055
SHKE 056
SHKE 057
SHKE 058
SHKE 059
SHKE 060
SHKE 061
SHKE 062
SHKE 063
SHKE 064
SHKE 065
SHKE 066
SHKE 067
SHKE 068
SHKE 069
SHKE 070

IF (IORIEN.EQ.0) II = M
IF (IORIEN.EQ.0 .AND. M.GT.MMAX) GO TO 60
IF (IORIEN.EQ.1) II=1
SINNU =SQRT((NY*NZI(II)-NZ*NYI(II))*2+(NZ*NYI(II)-NX*NZI(II))*2)
1  + (NX*NYI(II)-NY*NXI(II))*2)
NU = ARSIN(SINNU) * 0.5729578E02
NU = ABS(NU)
IF (NX.LT. NXI(II)) NU = -NU
ANGLE(2) = ABS(NU)

C
C SET UP DATA FOR COMPR OR EXPAND
DO 10 I=1,8
10 FS(I) = BSS(I,II)
ISDET = 0
IF (NU.GT.0.00001 .AND. NU.LT.0.00001) GO TO 30
IF (NU.LT.0.0) GO TO 20
CALL COMPR (ANGLE, MER, IPRCK, CPSTAG, ISDET, IFIRST, CP)
GO TO 50
20 CALL EXPAND (ANGLE, MER, IPRCK, ISDET, CP)
GO TO 50

C
30 DO 40 I=1,8
40 BS(I) = FS(I)
50 CP=(RS(2)/FSS(2)+1.0)/(0.5+GAM*FSS(6)+FSS(6))
GO TO 150

C
C*****
C***** CALCULATION FOR INITIAL ELEMENT OF EACH ROW *****
C*****
C
C CHECK IF THERE ARE TOO MANY INITIAL ELEMENTS (MAXIMUM OF 100)
60 IF ((IORIEN.EQ.0 .AND. M.GT. 30).OR.(IORIEN.EQ.1.AND.N.GT. 30))
1 GO TO 170

C
MMAX = M
```

SHKEXP

SHKEXP

```

C
IF (IORIEN.EQ.0) IISM
IF (IORIEN.EQ.1) IISN
IF (ISMODE .EQ. 1) GO TO 150

    ANGLE(2) = ARS(DELTA)
    NU = DELTA
    ISNET = 0

C
CHECK IF IMPACT OR SHADOW
IF (DELTAR .LE. 0.0) GO TO 110

C
IMPACT FLOW *****
GO TO (200,200,70,200,80,200,200,200,200,200,200,200,90), IMPACTI

C
TANGENT WEDGE
70 CALL COMPR (ANGLE, MER, IPACK, CP3TAG, ISDET, IFIRST, CP)
GO TO 150

C
TANGENT CONE EMPIRICAL
80 ANGLF(1) = DFLTA
CALL CONE (ANGLE, CP, 0)
GO TO 150

C
DELTA WING EMPIRICAL FOR SHOCK-EXPANSION CALCS.
90 DELDLW = DFLTAR
IF (DELDLW .LT. 0.01745) DELDLW = 0.01745
FMNS = FS(6) * SIN(DELDLW)
EMNS = 2. * (GAM+1.0) / (GAM+3.1) * EMNS * EXP((0.25 * (GAM+1.0)
1      * 2. * (GAM+1.0) / (GAM+3.1)) * EMNS)
CP = 4. / (FS(6) ** 2 * (GAM+1.0)) * (EMNS ** 2 - 1.0)
P2PII = 0.5 * GAM * FS(6) ** 2 * CP + 1.0
RS(2) = P2PII * FS(2)
TINTP = ((GAM+1.0) ** 2 * EMNS * EMNS) / ((2.0 * GAM * FMNS * EMNS * (GAM+1.0)) *
1      ((GAM+1.0) * EMNS * EMNS + 2.0))
RS(6) = SQRT((FS(6) ** 2 * (4.0 * (FMNS-1.0) * (GAM * FMNS + 1.0)) /
1      ((GAM+1.0) ** 2 * EMNS)) * TINTP)

```

SHKE 071
SHKE 072
SHKE 073
SHKE 074
SHKE 075
SHKE 076
SHKE 077
SHKE 078
SHKE 079
SHKE 080
SHKE 081
SHKE 082
SHKE 083
SHKE 084
SHKE 085
SHKE 086
SHKE 087
SHKE 088
SHKE 089
SHKE 090
SHKE 091
SHKE 092
SHKE 093
SHKE 094
SHKE 095
SHKE 096
SHKE 097
SHKE 098
SHKE 099
SHKE 100
SHKE 101
SHKE 102
SHKE 103
SHKE 104
SHKE 105

SHKEXP

SHKEXP

SHKE 106
 SHKE 107
 SHKE 108
 SHKE 109
 SHKE 110
 SHKE 111
 SHKE 112
 SHKE 113
 SHKE 114
 SHKE 115
 SHKE 116
 SHKE 117
 SHKE 118
 SHKE 119
 SHKE 120
 SHKE 121
 SHKE 122
 SHKE 123
 SHKE 124
 SHKE 125
 SHKE 126
 SHKE 127
 SHKE 128
 SHKE 129
 SHKE 130
 SHKE 131
 SHKE 132
 SHKE 133
 SHKE 134
 SHKE 135
 SHKE 136
 SHKE 137
 SHKE 138
 SHKE 139
 SHKE 140

```

BS(3) = FS(3) / TIM72
BS(1) = RM0F9 * (BS(2)/PFS) * (TFS/BS(3))
BS(4) = SQRT((GAM-1.)*GASC0*BS(3))
IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.6)
IF(BS(3).LT.AV1) BS(5) = AV2*BS(3)**AV2
BS(7) = BS(4) * BS(6)
BS(A) = BS(1) * BS(7) / BS(5)
GO TO 150
C SHADOW FLOW *****
110 GO TO (120,130,120,120,120,120,120),ISHADI
120 GO TO 200
C EXPANSION FROM PRESTREAM
130 ISDET = 0
CALL EXPAND (ANGLE,MEG,IPROK,ISDET,CP)
C
150 NXI(II) = NX
NYI(II) = NY
NZI(II) = NZ
151 DO 152 I=1,6
152 BSS(I,II) = BS(I)
C
IF (IPRINT .NE. 0) WRITE (TAPEOT,160) LL,N,M,BS(2),BS(3),BS(6),CP,
1 NU,PS(2),FS(3),FS(6)
160 FORMAT (1H,32H SHKEXP. LOCAL CONDITIONS LL=I5,4H N=I4,
1 4H M=I4,4H P=E12.5,4H T=E12.5,7H MACH=F7.3,5H CP=E12.5,/
2 1H ,28X,13HTURN ANGLE =F9.4,4X,3HPI=E12.5,4H TIME=I2.5,
3 7H MACH1=F7.3 )
DO 165 I=1,6
165 FS(I) = FSS(I)
RETURN
C
170 WRITE (TAPEOT,180)
180 FORMAT (1H ,48H*****NUMBER OF INITIAL ELEMENTS CANNOT EXCEED 100

```

SHKEXP

SHKEXP

```

1 57H FOR SHOCK-EXPANSION CALCULATIONS, CHANGE INPUT DATA*** )
  WRITE (TAPEOT,190) IRRLEN,LL,N,M
190 FORMAT (1H,10X,AH1ORIEN =I2,5X,
1 5X,4HLI =I5,5X,3HN =I5,5X,3HM =I5)
  ERROR = 3
  RETURN
200 WRITE (TAPEOT,210)
210 FORMAT (1H,47H***DURING SHOCK-EXPANSION CALCULATIONS PROGRAM
1 59H TRIED TO USE WRONG INITIAL ELEMENT METHOD--CHECK INPUT***)
  FRPDR = 3
  RETURN
  END

```

C

SHKE 141
SHKE 142
SHKE 143
SHKE 144
SHKE 145
SHKE 146
SHKE 147
SHKE 148
SHKE 149
SHKE 150
SHKE 151
SHKE 152
SHKE 153

SHKEXP

BLUNT

BLUN 001
 BLUN 002
 BLUN 003
 BLUN 004
 BLUN 005
 BLUN 006
 BLUN 007
 BLUN 009
 BLUN 009
 BLUN 010
 BLUN 011
 BLUN 012
 BLUN 013
 BLUN 014
 BLUN 015
 BLUN 016
 BLUN 017
 BLUN 018
 BLUN 019
 BLUN 020
 BLUN 021
 BLUN 022
 BLUN 023
 BLUN 024
 BLUN 025
 BLUN 026
 BLUN 027
 BLUN 028
 BLUN 029
 BLUN 030
 BLUN 031
 BLUN 032
 BLUN 033
 BLUN 034
 BLUN 035

SUBROUTINE BLUNT(PFS,MACH,TFS,VIS,RHOF,FB,RNO,TAU,IVISIN)
 C THIS SUBROUTINE CALCULATES THE VISCOUS FORCES ON A BLUNT
 C FACED BODY FOLLOWING THE APPROACH SUGGESTED BY L. GOLDBERG IN
 C G. E. REPORT R663021 (SEE ALSO R655050). THE STRAIGHT VISCOUS
 C FORCES ARE CALCULATED USING A SIMPLE CORRELATION FORMULA BASED ON
 C THE RESULTS OF SCALA AND GILBERT. THE LOW DENSITY OR VISCOUS
 C INTERACTION EFFECTS ARE BASED ON NUMERICAL RESULTS OF HIGHER
 C ORDER BOUNDARY-LAYER SOLUTIONS. THE SHEAR EFFECTS ARE A COMPLICATED
 C FUNCTION OF THE INVERSE DENSITY RATIO AND THE SHOCK REYNOLDS
 C NUMBER. IN THE PRESENT CALCULATIONS THESE LOW DENSITY
 C EFFECTS ARE DETERMINED FROM A SET OF EXPONENTIAL FUNCTION
 C CURVES WHICH HAVE BEEN MATCHED TO THE NUMERICAL RESULTS.
 C CALCULATION OPTIONS ARE CONTROLLED BY THE FLAG IVISIN,

IVISIN	GAS	VISCOUS=INTERACTION
0	IDEAL	NO
1	IDEAL	YES
2	REAL	NO
3	REAL	YES

THE LAST TWO OPTIONS ARE NOT YET AVAILABLE.
 *****D. N. SMYTH PROGRAM A1THOR*****

COMMON /GAS/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYP(2)
 COMMON /TAPE/TAPEIN,TAPECT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
 1 TAPEG,TAPFH,TAPEI,TAPEJ,TAPEK

REAL MACH
 INTEGER TAPEIN,TAPECT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
 1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
 DATA ADD, RFD, A1, B1, COK, XOEY, EVK, EPS /

BLUNT

BLUNT

```
1 1.0, 3.2907, 0.667, 1.1111, -2.0, -0.3, -1.80, 0.01 /
G = GAM
GCP = GASCP
GPI = G + 1.0
GMI = G - 1.0
C IF (IVISIN,LT,2) GO TO 10
C REAL GAS SOLUTION (TO BE ADDED).
C IDEAL GAS SOLUTION. ALL EQUATIONS FROM NACA TR-1135.
C INVERSE DENSITY RATIO ACROSS NORMAL SHOCK.
10 RORAI = (GMI + 2.0/MACH**2)/GPI
C TEMPERATURE BEHIND NORMAL SHOCK.
T2 = TFS*(2.0*G*MACH**2-GMI)*(GMI*MACH**2+2.)/(GPI*MACH)**2
C CALCULATE VISCOSITY.
IF (T2.LT,AV1) VIS2 = AV2*T2**AV3
IF (T2.GE,AV1) VIS2 = 2.27F-R*T2**1.5/(T2+198.6)
C REYNOLDS NUMBER BEHIND NORMAL SHOCK.
RES = RENO*PR*VIS/VIS2
C CALCULATE SHEAR COEFFICIENT.
20 CFC = 2.0/(SQRT(PES)*(1.0 - 0.495*SQRT(RORAI)))
C CHECK IF LOW DENSITY VISCOUS-INTERACTION EFFECTS DESIRED.
TMPL = 1.0
IF((IVISIN,EO,0).OR,(IVISIN,EO,2))GO TO 70
C DETERMINE LOW DENSITY EFFECTS. CALCULATE INDEPENDENT VARIABLE, EX.
FX = ALOG10(PES*RORAI**3)
C CHECK BOUNDARIES
```

BLUN 036
BLUN 037
BLUN 038
BLUN 039
BLUN 040
BLUN 041
BLUN 042
BLUN 043
BLUN 044
BLUN 045
BLUN 046
BLUN 047
BLUN 048
BLUN 049
BLUN 050
BLUN 051
BLUN 052
BLUN 053
BLUN 054
BLUN 055
BLUN 056
BLUN 057
BLUN 058
BLUN 059
BLUN 060
BLUN 061
BLUN 062
BLUN 063
BLUN 064
BLUN 065
BLUN 066
BLUN 067
BLUN 068
BLUN 069
BLUN 070

BLUNT

BLUNT

BLUN 071
 BLUN 072
 BLUN 073
 BLUN 074
 BLUN 075
 BLUN 076
 BLUN 077
 BLUN 078
 BLUN 079
 BLUN 080
 BLUN 081
 BLUN 082
 BLUN 083
 BLUN 084
 BLUN 085
 BLUN 086
 BLUN 087
 BLUN 088
 BLUN 089
 BLUN 090
 BLUN 091
 BLUN 092
 BLUN 093
 BLUN 094
 BLUN 095
 BLUN 096
 BLUN 097
 BLUN 098
 BLUN 099
 BLUN 100
 BLUN 101
 BLUN 102
 BLUN 103
 BLUN 104
 BLUN 105

```

IF (EX.GT.3.0) GO TO 70
TWBL = 0.0
IF (EX.LT.6.0) GO TO 70

C
C
30 F1 = A1 - B1*EX
IF (EX.GT.3.0) GO TO 40
Y2 = F1
GO TO 60

C
C
40 DXFV = EX * XOEY
IF (ABS(DXEY).LT.EPS)GO TO 50
Y2 = F1 + (.1.0 * F1)/(1.0 - EXP(EVK*DXFV))
GO TO 60

C
C
50 Y2 = F1 + B1/(EVK*(1.0 + 0.5*EVK*DXEY));
60 DXND = EX * (AND + BDD*ALOG10(RORAI))
TWBL = Y2/(1.0 + EXP(DOK*DXND));

C
C
70 TAU = TWBL*CFD

C
C
THE FOLLOWING CARDS ARE FOR CHECKOUT ONLY (SET IPRINT = 1),
IPRINT = 0

C
IF (IPRINT.NE.1) RETURN
WRITE (TAPEOT,60) RORAI,RES,EX,CFD,TAU
80 FORMAT(IH1, 2X7HRORAI =,E13.6, 5X5HRES =,E13.6, 5X11H(E**3)RES =,
1 E13.6, 5X5HCFD =,E13.6, 5X5HTAU =,E13.6)
WRITE (TAPEOT,90) PFS,TFS,RHOF9,VIS,REND
90 FORMAT(IH0,2X7H PFS =,E13.6, 5X5HTFS =,E13.6, 5X11H RHOF9 =,

```

BLUNT

BLUNT

F13,6,5X5HVIS 3,E13,6, 4X6HRENO 3,E13,6)

1 RETURN
END

BLUN 106
BLUN 107
BLUN 108

BLUNT

CPINPT

```
C
C
SURROUTINE CPINPT(IMP, LINTAB)
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /GDATA/LTOT, J, SYMFCT, ICRN, IGTYPF, L
COMMON /ABDATA/NAB, ALPHA(20), BETA(20), ROL(20), DELTA(20), GI(20),
1 RI(20), PI(20)
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF, TAPEG,
1 TAPEH, TAPEI, TAPEJ, TAPFK
COMMON /SURFN2/INORM, ISURF, IX, IY, IZ, UI, UL, VI, VL, CR, CT, CHX, N2, N3,
1 XO, YO, ZO, AP11, AP12, AP13, AP21, AP22, AP23, AP31, AP32, AP33,
2 FLOWC(7), XI(500), YI(500), B(503,7), XKF(503), AA(5999),
A XB(4), YB(4), ZB(4),
3 NDSET, IABSET, IR, LORG(20), ISR(20), NS, ISFR, IVIS, IFTYP, IFLOW,
4 RX(2,20), RY(2,20), NB
DIMENSION ELEM(25), LCOND(3), DIMFL(6)
DIMENSION TITLE(15), INP(20), LINTAB(20)
DIMENSION TITL(10), IMTAB(9), LOAB(20), TITLES(10), TITLEA(10),
1 TITLER(10), E4(12), E2(13), DAT(6),
3 E3(17), ISF(5), FDATA(34), XP(4,6)
INTEGER ERROR, PAGE, CASE, SYMFCT
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
1 TAPFG, TAPFH, TAPEI, TAPEJ, TAPEK
EQUIVALENCE (FDATA(1), LCOND(1)), (FDATA(4), FLEM(1)),
1 (FDATA(29), DIMFL(1))
DATA RC/0, 1743329E-1/
C FOR CHECKOUT SET IPRINT = 1
IPRINT = 0
C
C
WRITE(TAPEOT,5)
5 FORMAT(1H1,45HPREVIOUSLY SAVED SURFACE DATA WILL BE USED TO,
```

CPIN 001
CPIN 002
CPIN 003
CPIN 004
CPIN 005
CPIN 006
CPIN 007
CPIN 008
CPIN 009
CPIN 010
CPIN 011
CPIN 012
CPIN 013
CPIN 014
CPIN 015
CPIN 016
CPIN 017
CPIN 018
CPIN 019
CPIN 020
CPIN 021
CPIN 022
CPIN 023
CPIN 024
CPIN 025
CPIN 026
CPIN 027
CPIN 028
CPIN 029
CPIN 030
CPIN 031
CPIN 032
CPIN 033
CPIN 034
CPIN 035

CPINPT

CPINPT

```
1 17M CALCULATE FORCES)
C C BEGIN ALPHA=BETA CYCLE
  REWIND TAPE
  DO 2000 IAB = 1,NAB
  ICT = 0
  ISYM = 1
  IF (INP(IAB) .EQ. 0) GO TO 2000
C C INPUT REGION ID INFO AND NORMALIZATION FLAGS
  10 READ (TAPEIN,20) LASTR,NDSET,IABSET,IR,INORM,ISURF,IPF,
  1 (ISR(I),I=1,10), ISF
  20 FORMAT (I1,3I2,3I1, 10I2,5I1)
C C WRITE (TAPEOT,32) LASTR,NDSET,IABSET,IR,INORM,ISURF,
  1 IPF, (ISR(I),I=1,10), ISF
  32 FORMAT (I10,5I,6HLASTR=,I2,4X,6HNDSER=,I2,4X,7HIABSETS,I2,4X,
  1 4HIR =,I2,
  2 /1H , 4HIPFS, I2, 4X4HISR=, 10I3, 4X4HISFS, 5I2/)
C C READ BOUNDARY POINTS FOR NORMALIZATION
  DO 3A I=1,4
  3A READ (TAPEIN,50) XB(I),YB(I),ZB(I)
  50 FORMAT (3F10,0)
C C READ IN MASTER DIRECTORY
  ITAG10 = 1
  CALL READMS (10,TITLEM,10,ITAG10)
  READ(10,ITAG10) TITLFM
  ITAG10 = 2
  CALL READMS (10,IMTAB,9,ITAG10)
  READ(10,ITAG10) IMTAB
C C CHECK DATA SET NUMBER
  IF (IMTAB(1) .GE. NDSET) GO TO 40
```

CPINPT

CPINPT

```
WRITE (TAPEOT,39) NDSET,INTAB(1)
39 FORMAT (1H0,9H*NDSET =,12,31H IS GREATER THAN THE NUMBER OF .
1 4GHDATA SETS ACTUALLY ON UNIT 10. PROGRAM HALT. )
STOP
C
C
C READ IN FLOW DATA SET DIRECTORY
40 ITAG10 = INTAB(NDSET + 4)
CALL READMS (10,E4,12,ITAG10)
C
C READ (10*ITAG10) E4
DO 41 I=1,10
41 YTTLES(I) = F4(I)
MACH = E4(11)
NARS = E4(12)
ITAG10 = ITAG10 + 1
CALL READMS (10,LOAB,20,ITAG10)
READ(10*ITAG10) LOAB
C
C
C READ IN FLOW REGION DIRECTORY FOR REQUESTED ALPHA-BETA SET
ITAG10 = LOAB(IABSET)
CALL READMS (10,E2,13,ITAG10)
C
C READ (10*ITAG10) E2
DO 42 I=1,10
42 YTTLEA(I) = F2(I)
ALPHAS = E2(11)
BETAS = E2(12)
NREG = E2(13)
ITAG10 = ITAG10 + 1
CALL READMS (10,LORG,20,ITAG10)
READ(10*ITAG10) LORG
C
C
C
C
C READ IN REQUESTED FLOW REGIONS
```

CPIN 071
CPIN 072
CPIN 073
CPIN 074
CPIN 075
CPIN 076
CPIN 077
CPIN 078
CPIN 079C
CPIN 080I
CPIN 081
CPIN 082
CPIN 083
CPIN 084
CPIN 085
CPIN 086C
CPIN 087I
CPIN 088
CPIN 089
CPIN 090
CPIN 091C
CPIN 092I
CPIN 093
CPIN 094
CPIN 095
CPIN 096
CPIN 097
CPIN 098
CPIN 099C
CPIN 100I
CPIN 101
CPIN 102
CPIN 103
CPIN 104
CPIN 105

CPINPT

CPINPT

CPIN 106
CPIN 107
CPIN 108
CPIN 109
CPIN 110C
CPIN 111I
CPIN 112
CPIN 113
CPIN 114
CPIN 115
CPIN 116
CPIN 117
CPIN 118
CPIN 119
CPIN 120
CPIN 121
CPIN 122
CPIN 123C
CPIN 124I
CPIN 125
CPIN 126
CPIN 127
CPIN 128
CPIN 129
CPIN 130
CPIN 131
CPIN 132
CPIN 133
CPIN 134
CPIN 135
CPIN 136
CPIN 137
CPIN 138
CPIN 139
CPIN 140

```
C      ITAG10 = LOGG(IR)
C      CALL READMS (10,E3,17,ITAG10)
C      READ (10,ITA:10) E3
C      DO 91 I=1,10
C      91 TITLFR(I) = F3(I)
C      DO 92 I=1,5
C      92 IDTYP(I) = E3(I+10)
C      CONTINUE
C      IF TYP = IDTYP(1)
C      IFLOW = IDTYP(2)
C      IVIS = 0
C      IF (IFTYP .EQ. 4) IVIS = 1
C      94 NSREG = E3(14)
C      ITAG10 = ITAG10 + 1
C      CALL READMS (10,DAT, 6,ITAG10)
C      READ (10,ITAG10) DAT
C      XO = DAT( 1)
C      YO = DAT( 2)
C      ZO = DAT( 3)
C      PSIO = DAT( 4)
C      THFTO = DAT( 5)
C      PHIO = DAT( 6)
C      SINT = SIN(THETO*RC)
C      COST = COS(THETO*RC)
C      COSPS = COS(PSIO*RC)
C      SINPS = SIN(PSIO*RC)
C      SINP = SIN(PHIO*RC)
C      COSP = COS(PHIO*RC)
C      AP11 = COST*COSPS
C      AP12 = COST*SINPS
C      AP13 =--SINT
```

CPINPT

CPINPT

```

AP21 = COSP*SINPS + SINP*SINT*COSSPS
AP22 = COSP*COSSPS + SINP*SINT*SINPS
AP23 = SINP*COST
AP31 = SINP*SINPS + COSP*SINT*COSSPS
AP32 = SINP*COSSPS + COSP*SINT*SINPS
AP33 = COSP*COST

```

C

```

C SET INDICES FOR NORMALIZING DATA
IF(INORM .LE. 0) GO TO 100
GO TO (110,130,140,150),INORM

```

C

```

C NORMALIZE W.R.T. A,R
100 IX = 4
    IY = 5
    IZ = 6
    GO TO 190

```

C

```

C NORMALIZE W.R.T. X,Y
110 IX = 1
    IY = 2
    IZ = 3
    GO TO 190

```

C

```

C NORMALIZE W.R.T. X,Z
130 IX = 1
    IY = 3
    IZ = 2
    GO TO 190

```

C

```

C NORMALIZE W.R.T. Y,Z
140 IX = 2
    IY = 3
    IZ = 1
    GO TO 190

```

C

CPIN 141
CPIN 142
CPIN 143
CPIN 144
CPIN 145
CPIN 146
CPIN 147
CPIN 148
CPIN 149
CPIN 150
CPIN 151
CPIN 152
CPIN 153
CPIN 154
CPIN 155
CPIN 156
CPIN 157
CPIN 158
CPIN 159
CPIN 160
CPIN 161
CPIN 162
CPIN 163
CPIN 164
CPIN 165
CPIN 166
CPIN 167
CPIN 168
CPIN 169
CPIN 170
CPIN 171
CPIN 172
CPIN 173
CPIN 174
CPIN 175

CPINPT

CPIINPT

```
C NORMALIZE W.R.T. ALPHA
150 IX = 4
    IY = 6
    IZ = 5
    GO TO 190

C CALCULATE NORMALIZING LENGTHS
190 CONTINUE
    ND 95 I = 1.4
    XX = XB(I) * XO
    YY = YB(I) * YO
    ZZ = ZB(I) * ZO
    XP(I,1) = XX*AP11 + YY*AP12 + ZZ*AP13
    XP(I,2) = XX*AP21 + YY*AP22 + ZZ*AP23
    XP(I,3) = XX*AP31 + YY*AP32 + ZZ*AP33
    XP(I,4) = XP(I,1)
    XP(I,5) = SORT(XP(I,2)**2 + XP(I,3)**2)
    IF (XP(I,2) .EQ. 0.0) GO TO 194
    XP(I,6) = ATAN2(XP(I,2), -XP(I,3))
    GO TO 95

194 XP(I,6) = 0.0
    IF (XP(I,3) .GT. 0.0) XP(I,6) = 3.141592654
    95 CONTINUE

C
    U1 = XP(1,IX)
    UL = XP(2,IX) * U1
    CR = UL
    IF (ISURF .EQ. 1) GO TO 96
    V1 = XP(3,IX)
    VL = XP(4,IX) * V1
    GO TO 97

C
    96 UL = XP(3,IX)
    CT = XP(4,IX) * XP(3,IX)
```

CPIINPT

CPINPT

CPIN 211
 CPIN 212
 CPIN 213
 CPIN 214
 CPIN 215
 CPIN 216
 CPIN 217
 CPIN 218
 CPIN 219
 CPIN 220
 CPIN 221
 CPIN 222
 CPIN 223
 CPIN 224
 CPIN 225
 CPIN 226
 CPIN 227
 CPIN 228
 CPIN 229
 CPIN 230
 CPIN 231
 CPIN 232
 CPIN 233
 CPIN 234
 CPIN 235
 CPIN 236
 CPIN 237
 CPIN 238
 CPIN 239
 CPIN 240
 CPIN 241
 CPIN 242
 CPIN 243
 CPIN 244
 CPIN 245

```

    C      VI = XP(1,IV)
    C      VL = XP(3,IV) + VI
    C      97 CONTINUE
    C
    C      ESTABLISH SUB-REGION COUNTERS
    C      IF (.SR(I) .GT. 0) GO TO 210
    C      NS = NSREG
    C      DO 200 I = 1,NS
    C      200 ISR(I) = I
    C      GO TO 230
    C      210 NS = 0
    C      DO 220 I = 1,10
    C      IF (ISR(I) .LE. 0) GO TO 230
    C      220 NS = NS + 1
    C      230 IF (NS .LE. NSREG) GO TO 250
    C      WRITE(TAPEOT,240) NS,NSREG
    C      240 FORMAT(1H0,28HSURFACE INTERPOLATION, THE ,I2,13H SUB-REGIONS ,
    C      1 36HREQUESTED IS GREATER THAN AVAILABLE./1H0, 12HCOUNTER SET ,
    C      2 11HTO NSREG = , I2, 20H AND CASE CONTINUES.)
    C      NS = NSREG
    C      250 CONTINUE
    C
    C      REWIND TAPEE
    C      INTAPE = TAPEC
    C      ITAPES = TAPED
    C
    C      I = 0
    C      ISFR = 0
    C      ICYCLE = 0
    C      NTOT = 0
  
```

CPINPT

CPINPT

```
IF (JPF .NE. 0) GO TO 1520
1100 CONTINUE
ERROR = 0
CALL SFNTR2
IF (ERROR .EQ. 1) GO TO 1520

C
C START CYCLE ON THE ELEMNTS
1110 REWIND INTAPE
REWIND ITAPES
ISAVET = ITAPES
ICYCLE = ICYCLE + 1
IF (IPRINT .NE. 1) GO TO 1111
WRITE(TAPEOT,9030) INTAPE,ITAPES,LTOT
9030 FORMAT(1HG,23HAT LOOP 1500, INTAPE = ,13, 5X9HITAPES = ,13,
1 SX7HLTOT = ,15)
1111 CONTINUE
DO 1500 II = 1,LTOT
IF (ICYCLE .GT. 1) GO TO 1050
READ(TAPEE,END=1510) IG4,ELEM
READ(TAPEE) IG4,ELEM
IF (FOF(TAPEE)) 1510,1012
1012 LCOND(2) = ELEM(1)
GO TO 1090
1050 CONTINUE
C READ(INTAPE,END=1510) IG4,FDATA
READ(INTAPE) IG4,FDATA
C IF (FOF(INTAPE)) 1510,1080
C
C CHECK IF ELEMENT ALREADY PROCESSED ON PREVIOUS CYCLE
1080 IF (LCOND(1) .NE. 0) GO TO 1200
1090 CONTINUE
IF (ISYM .EQ. 1) GO TO 1095
C REFLECT ELEMENT TO -Y SIOF AND CHANGE ORDER OF POINTS 2 AND 4
IG4 = -IG4
ELEM(5) = -ELEM(5)
```

CPINPT

CPINPT

ELEM(8) = ELEM(8)
 ELEM(15) = ELEM(15)
 ELEM(16) = ELEM(16)
 ELEM(17) = ELEM(17)
 ELEM(18) = ELEM(18)
 PLT = ELEM(12)
 ELEM(12) = ELEM(14)
 ELEM(14) = ELT
 ELT = ELEM(16)
 ELEM(16) = ELEM(18)
 ELEM(18) = ELT
 FLY = ELEM(20)
 ELEM(20) = ELEM(22)
 ELEM(22) = ELT

CPIN 281
 CPIN 282
 CPIN 283
 CPIN 284
 CPIN 285
 CPIN 286
 CPIN 287
 CPIN 288
 CPIN 289
 CPIN 290
 CPIN 291
 CPIN 292
 CPIN 293
 CPIN 294
 CPIN 295
 CPIN 296
 CPIN 297
 CPIN 298
 CPIN 299
 CPIN 300
 CPIN 301
 CPIN 302
 CPIN 303
 CPIN 304
 CPIN 305
 CPIN 306
 CPIN 307
 CPIN 308
 CPIN 309
 CPIN 310
 CPIN 311
 CPIN 312
 CPIN 313
 CPIN 314
 CPIN 315

C INTERPOLATE FOR SURFACE PROPERTIES AT CENTROIDS.
 C 1095 CONTINUE

IC = 0
 IT = 1
 CALL VALU2(1, ELEM(7), ELEM(8), ELEM(9), XN, YN, ZN, IC, IT, INP,
 1 DINF(1), DINF(2), DINF(3), DINF(4), DINF(5), DINF(6))

C
 LCOND(1) = 0
 IF (INR.EQ. 0) GO TO 1200
 LCOND(1) = 1
 NTOT = NTOT + 1

C
 C SAVE RESULTS ON TEMPORARY UNIT
 C 1200 CONTINUE
 WRITE(ITAPES) I64,FOATA

C
 C GO TO NEXT ELEMENT
 C 1500 CONTINUE
 C 1510 CONTINUE
 C

CPINPT

CPINPT

```

C 1520 I = Y + 1
      IF (I .GT. 5) GO TO 1600
      ISFR = ISF(I)
      IF (IPRINT .NE. 1) GO TO 1525
      WRITE(TAPEOT,9010) I,ISFR,ICYCLE,NTOT,ERROR
9010  FORMAT(IH0, 4H1 = ,I3, 5X7HISFR = ,I3, 5X9HICYCLE = ,I3,
      1 7HNTOT = ,I5, 5X4HERROR = ,I3)
1525  CONTINUE
      IF (ISFR .EQ. 0) GO TO 1600
      IF (ICYCLE .EQ. 0) GO TO 1100
      IF (NTOT .EQ. LTOT) GO TO 1600

C SWITCH TAPES (ONLY IF ERROR = 0)
C  IF (ERROR .NE. 0) GO TO 1100
      I1 = ITAPES
      ITAPFS = INTAPE
      INTAPE = I1
      GO TO 1100

C SAVE RESULTS ON TAPE
C 1600 CONTINUE
      IF (ICYCLE .GT. 0) GO TO 1560
      WRITE(TAPEOT,1550)
1550  FORMAT(IH1,47HSURFACE INTERPOLATION (CPINT) CANNOT BE MADE./
      154HOCHECK THAT PROPER FLOW FIELD DATA HAS BEEN SPECIFIED.,
      2 14H RUN IS STOPPED.)
      STOP

C 1560 REWIND ISAVET
      IF (IPRINT .NE. 1) GO TO 1601
      WRITE(TAPEOT,9020) L,NT,INTAPE,ISAVET
9020  FORMAT(IH0, 7H1TOT = ,I5, 5X9HINTAPE = ,I3, 5X9HISAVET = ,I3)
1601  CONTINUE
      DO 1610 L = 1,LTOT
```

CPIN 316
CPIN 317
CPIN 318
CPIN 319
CPIN 320
CPIN 321
CPIN 322
CPIN 323
CPIN 324
CPIN 325
CPIN 326
CPIN 327
CPIN 328
CPIN 329
CPIN 330
CPIN 331
CPIN 332
CPIN 333
CPIN 334
CPIN 335
CPIN 336
CPIN 337
CPIN 338
CPIN 339
CPIN 340
CPIN 341
CPIN 342
CPIN 343
CPIN 344
CPIN 345
CPIN 346
CPIN 347
CPIN 348
CPIN 349
CPIN 350

CPINPT

CPINPT

```
C READ(ISAVET,END=1620) IG4,FDATA
  READ(ISAVET) IG4,FDATA
  IF (EOF(ISAVET)) 1620,1605
1605 WRITE(TAPEF) IG4,FDATA,NDSET,IARSET,NS,IPF,ISPR
     ICT = ICT + 1
     IF (IPRINT.NE.1) GO TO 1610
     WRITE(TAPECT,9000) IG4,LCOND,FLEM,DINFL
9000 FORMAT(1M0,6MIG4 = ,15,5X8HLCOND = ,3J3/
     1  4M ELEM = ,10E12,4/1M ,7X10E12,4/1M , 7X,5E12,4/
     2  4M DINFL = , 6E12,4)
1610 CONTINUE
1620 CONTINUE
C
C CHECK FOR SYMMETRY REQUIREMENTS
  IF (SYMFCT.EQ.1 .OR. BETAS.EQ.0.0 .OR. ISYM.EQ.2) GO TO 1900
  REWIND TAPEE
  ISYM = 2
  GO TO 10
C
1900 LTTAB(IA8) = ICT
C END OF ALPHA=BETA LOOP
2000 CONTINUE
C
  REWIND TAPEE
  REWIND TAPEF
C
  RETURN
  END
```

CPIN 3511
CPIN 3520
CPIN 3530
CPIN 354
CPIN 355
CPIN 356
CPIN 357
CPIN 358
CPIN 359
CPIN 360
CPIN 361
CPIN 362
CPIN 363
CPIN 364
CPIN 365
CPIN 366
CPIN 367
CPIN 368
CPIN 369
CPIN 370
CPIN 371
CPIN 372
CPIN 373
CPIN 374
CPIN 375
CPIN 376
CPIN 377
CPIN 378

CPINPT

```

SURROUTINE      FLNTRP
C
COMMON /EXEC/CASF, TITLF, PAGE, ERROR
COMMON /GDATA/LTOT, JJ, SYMFCT, IORN, IGTYPE, LL
COMMON /TAPE/TAPEIN, TAPEIN, TAPEIN, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /ABDATA/NAB, ALPHA(20), BETA(20), GAMMA(20), DELTA(20), Epsilon(20),
      RI(20), PI(20)
COMMON /INTERF/INT, ISWE(20), NSWE(100), Y9, ISWFF, INF(20,22),
      DINF(20,6), LNTTAB(20)
INTEGER ERROR, SYMFCT, PAGF, CASE
INTEGER TAPEIN, TAPEIN, TAPEIN, TAPEB, TAPEC, TAPEE, TAPEF, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
DIMENSION ELEM(25), XIN(4), YIN(4), ZIN(4)
DIMENSION TITLE(15), TITLEM(10), TITLFS(10), TITLEA(10),
      TITLER(10)
DIMENSION IFL(4), INTAB(9), LOAB(20), LOPG(20), LOFF(5), LOCD(5),
      IOTYP(5), IFC(4,6), ITFLAG(4), XU(4), YO(4), ZO(4),
      PSI0(4), THETA(4), PHIO(4), CNST(4), SINT(4), COSS(4),
      SINS(4), IX(4), IY(4), AI(4), A2(4),
      XI(103), YI(103), FI(103), XB(4,100), XX(4,103), YY(4,103),
      COORDS(6), FLOW(103,6), XKF(103), A( 5429), B(103,6),
      CFB(4,2,53), CFF(4,103,6), PB(2), XPF(6), LCOND(3),
      DINFL(6), DAT( 6), FIC(4,2), FLOWC(4,6), FI(25), E2(13),
      E3(17), E4(12), E5(25), E6(6), E7(3),
      ISR(4), ISF(4), IFD(4,6), LOSF(5)
COMMON/SURFEN2/WK(11197)
EQUIVALENCE (WK(1), XI(1)), (WK(104), YI(1)), (WK(207), FI(1)),
      (WK(310), XX(1,1)), (WK(722), YY(1,1)), (WK(1134), FLOW(1,1)),
      (WK(1752), B(1,1)), (WK(2370), XKF(1)), (WK(2473), CFF(1,1,1)),
      (WK(4945), XB(1,1)), (WK(5345), CFB(:,1,1)), (WK(5769), A(1))
REAL MACH
DATA RC, NX, MY, MX1, 0, K0
      /0.1745329E-1, 103, 6, 1, 1.0, 2000/

```

```

FLNT 001
FLNT 002
FLNT 003
FLNT 004
FLNT 005
FLNT 006
FLNT 007
FLNT 008
FLNT 009
FLNT 010
FLNT 011
FLNT 012
FLNT 013
FLNT 014
FLNT 015
FLNT 016
FLNT 017
FLNT 018
FLNT 019
FLNT 020
FLNT 021
FLNT 022
FLNT 023
FLNT 024
FLNT 025
FLNT 026
FLNT 027
FLNT 028
FLNT 029
FLNT 030
FLNT 031
FLNT 032
FLNT 033
FLNT 034
FLNT 035

```

FLNT 036
 FLNT 037
 FLNT 038
 FLNT 039
 FLNT 040
 FLNT 041
 FLNT 042
 FLNT 043
 FLNT 044
 FLNT 045
 FLNT 046
 FLNT 047
 FLNT 048
 FLNT 049
 FLNT 050
 FLNT 051
 FLNT 052
 FLNT 053
 FLNT 054
 FLNT 055
 FLNT 056
 FLNT 057
 FLNT 058
 FLNT 059
 FLNT 060
 FLNT 061
 FLNT 062
 FLNT 063
 FLNT 064
 FLNT 065
 FLNT 066
 FLNT 067
 FLNT 068
 FLNT 069
 FLNT 070

```

    TFX(XO,X,Y,Z) = XO + AP11*X + AP21*Y + AP31*Z
    TFY(YO,X,Y,Z) = YO + AP12*X + AP22*Y + AP32*Z
    TFZ(ZO,X,Y,Z) = ZO + AP13*X + AP23*Y + AP33*Z

    C REWIND TAPEF
    C CYCLE ON ALPHA, BETA
    DO 2000 IAB = 1,NAB

    C DATA SET NUMBER AND ALPHA-BETA SET NUMBER
    NDSET = INF(IAB,2)
    IARSET = INF(IAB,3)

    C WRITE(TAPEOT,3000) IAB, NDSET, (INF(IAR,I),I=1,10)
    3000 FORMAT(1H1,30HFLOW INTERPOLATION SUBROUTINE /1H0,
    1 6HIAB = ,I3, 5X 8HNDSET = ,I3,5X6HINF = ,10(5X13))
    WRITE(TAPEOT,3010) ALPHA(IAB), BETA(IAB)
    3010 FORMAT(1H0, 8HALPHA = , F10.6, 7HBETA = , F10.6)

    C DEFINE FLOW FIELD REGIONS
    NFR = 0
    J = 0
    DO 10 I = 1,4
    J = J + 3
    IF (INF(IAB,J) .EQ. 0) GO TO 20
    NFR = NFR + 1
    IFL(NFR) = INF(IAB,J)
    ISR(NFR) = INF(IAB,J+1)
    ISF(NFR) = INF(IAB,J+2)
    10 CONTINUE

    C CHECK IF DEFAULT TO FREE STREAM
    20 IF (NFR .EQ. 0) GO TO 1010
    WRITE(TAPEOT,3020) NFR, (IFL(NR), NR=1,NFR)
    3020 FORMAT(1H0,35HNUMBER OF FLOW REGIONS REQUESTED = ,I3, 10X,
    1 4(5X13))

    C
    
```

FLNTRP

FLNT 071
 FLNT 072
 FLNT 073C
 FLNT 074I
 FLNT 075
 FLNT 076C
 FLNT 077I
 FLNT 078
 FLNT 079
 FLNT 080
 FLNT 081
 FLNT 082
 FLNT 083
 FLNT 084
 FLNT 085
 FLNT 086C
 FLNT 087I
 FLNT 088
 FLNT 089
 FLNT 090
 FLNT 091
 FLNT 092
 FLNT 093C
 FLNT 094I
 FLNT 095
 FLNT 096
 FLNT 097
 FLNT 098C
 FLNT 099I
 FLNT 100
 FLNT 101
 FLNT 102
 FLNT 103
 FLNT 104
 FLNT 105

```

C READ IN MASTER DIRECTORY
  ITAG10 = 1
  CALL READMS (10, TITLEM, 10, ITAG10)
  READ(10:ITAG10) TITLEM
  IYAG10 = 2
  CALL READMS (10, IMTAB, 9, ITAG10)
  READ(10:ITAG10) IMTAB

C CHECK DATA SET NUMBER
  IF (IMTAB(1) .GE. NDSSET) GO TO 40

C
C
C READ IN FLOW DATA SET DIRECTORY
  40 ITAG10 = IMTAB(NDSSET + 4)
  CALL READMS (10, E4, 12, ITAG10)
  READ (10:ITAG10) E4
  DO 41 I=1,10
  41 TITLES(I) = E4(I)
  MACH = E4(11)
  NARS = E4(12)
  ITAG10 = ITAG10 + 1
  CALL READMS (10, LOAD, 20, ITAG10)
  READ(10:ITAG10) LOAD

C
C READ IN FLOW REGION DIRECTORY FOR REQUESTED ALPHA BETA SET
  ITAG10 = LOAD(IARSET)
  CALL READMS (10, E2, 13, ITAG10)
  READ (10:ITAG10) E2
  DO 42 I=1,10
  42 TITLEA(I) = F2(I)
  ALPHAS = E2(11)
  RETAS = E2(12)
  NRFG = E2(13)
  ITAG10 = ITAG10 + 1
  
```

FLNTRP

FLNTRP

FLNT 106C
FLNT 107I
FLNT 108
FLNT 109
FLNT 110
FLNT 111
FLNT 112
FLNT 113
FLNT 114
FLNT 115
FLNT 116
FLNT 117
FLNT 118
FLNT 119
FLNT 120
FLNT 121
FLNT 122
FLNT 123
FLNT 124
FLNT 125
FLNT 126
FLNT 127
FLNT 128
FLNT 129
FLNT 130
FLNT 131
FLNT 132
FLNT 133
FLNT 134
FLNT 135
FLNT 136
FLNT 137C
FLNT 138I
FLNT 139
FLNT 140

```
      CALL READMS (10,LORG,20,ITAG10)  
      READ(10,ITAG10) LORG  
      C  
      C CHECK IF STORED A=B SET CONSISTENT WITH REQUEST  
      IF (ALPHA(IAB) .NE. ALPHAS) GO TO 50  
      IF (BETA(IAB) .EQ. BETAS) GO TO 50  
      C  
      C A=B NOT CONSISTENT  
      50 STOP  
      C  
      C  
      C READ IN REQUESTED FLOW REGIONS  
      C FIRST CHECK IF REQUESTED REGIONS AVAILABLE  
      60 CONTINUE  
      DO 70 NR = 1,NFR  
      IF (IFL(NR) .GT. NREG) GO TO 80  
      70 CONTINUE  
      GO TO 90  
      C  
      C  
      80 WRITE(6, 85) NR, NDSET  
      85 FORMAT(1H, 26HREQUESTED FLOW REGION NO. ,12, 12H OF DATA SET, 13,  
      1 15H NOT AVAILABLE.)  
      1 STOP  
      C  
      C READ IN FLOW FIELD REGIONS AND CALCULATE COEFFICIENTS  
      90 CONTINUE  
      DO 1000 NR = 1,NFR  
      IR = IFL(NR)  
      ITAG10 = LORG(IR)  
      CALL READMS (10,E3,17,ITAG10)  
      READ (10,ITAG10) E3  
      DO 91 I=1,10  
      91 TITLER(I) = F3(I)
```

FLNTRP

FLNTRP

FLNT 141
FLNT 142
FLNT 143
FLNT 144
FLNT 145
FLNT 146
FLNT 147
FLNT 148
FLNT 149
FLNT 150
FLNT 151
FLNT 152
FLNT 153
FLNT 154
FLNT 155
FLNT 156C
FLNT 157I
FLNT 158
FLNT 159
FLNT 160
FLNT 161
FLNT 162
FLNT 163
FLNT 164
FLNT 165
FLNT 166
FLNT 167
FLNT 168
FLNT 169C
FLNT 170I
FLNT 171
FLNT 172
FLNT 173
FLNT 174
FLNT 175

```
DO 92 I=1,5
  IDTYP(I) = E3(I+10)
92 CONTINUE
  NSREG = E3(16)
  IFLAG(NR) = E3(17)

C
C CHECK FLOW-FIELD DATA TYPE FLAG
  IF (IDTYP(1) .EQ. 1) GO TO 98
  WRITE (TAPE,97)
97 FORMAT (1H0,45H**IN ATTEMPTING TO READ FLOW FIELD DATA FROM ,
1 SPHINT 10 IDTYP(1) WAS NOT EQUAL TO 1 AS IT SHOULD BE.,
2 14H PROGRAM STOP. )
  STOP
98 IF (IDTYP(2) .EQ. 1) GO TO 1600
  ITAG10 = ITAG10 + 1
  CALL READMS (10,DAT, 6,ITAG10)
  READ (10*ITAG10) DAT
  XO(NR) = DAT( 1)
  YO(NR) = DAT( 2)
  ZO(NR) = DAT( 3)
  PSTO(NR) = DAT( 4)
  THFTO(NR) = DAT( 5)
  PHTO(NR) = DAT( 6)

C
C LOCATE SUR-REGION
  IS = 0
  IG10 = LORG(IR) + 4 + ISR(NR)
93 CONTINUE
  CALL READMS(10, E1, 25, IG10)
  READ(10*IG10) E1
  DO 94 I = 1,5
    LOFF(I) = E1(I)
    IFF(NR,I) = F1(I+5)
    LOCD(I) = E1(I+10)
    YFR(NR,I) = F1(I+15)
```

FLNTRP

FLNTRP

```

          94 CONTINUE
          IF (IS,NE,0) GO TO 96
          DO 95 I = 1,5
          95 LOSF(I) = E1(I+20)

          C CHECK IF SECONDARY FLOW
          IF (ISF(NR) .EQ. 0) GO TO 96
          IS = ISF(NR)
          IG10 = LOSF(IS)
          GO TO 93

          C
          C
          96 CONTINUE
          WRITE(TAPEOT,3030) TITLE, IMTAB, TITLES, MACH, NABS, LOAB,
          1 TITLE, ALPHAS, BETAS, NREG, LORG, TITLER, IDTYP, LOFF
          3030 FORMAT(1H, 7HTITLE=,10A4, 7H IMTAB=, 9I4, /1H, 7HTITLES=,10A4,
          1 7H MACH =, F6.3, 6H NABS=,13, / 1H, 4X, 5H LOAB, 20I4, /1H, 7HTITLE=,
          210A4, 8H ALPHAS=,F6.3, 7H BETAS=,F6.3, 8H NREG =, I3, /1H, 4X,
          3 5H LORG=, 20I4, /1H, 7HTITLER=,10A4, 7H IDTYP=,5I2, 7H LOFF =,5I4)
          WRITE(TAPEOT,3040) (IFC(NR,I),I=1,5), ITFLAG(NR), XO(NR),
          1 YO(NR), ZO(NR), PSIO(NR), THETO(NR), PHIO(NR)
          3040 FORMAT(1H, 6H IFC =, 5I5, 9H ITFLAG =, I3.6H XO =, F10.6,
          1 6H YO =, F10.6, 6H ZO =, F10.6/1H,
          2 8H PSIO =, F10.6, 10H THETO =, F10.6, 9H PHIO =, F10.6)
          C SFT INDICES TO BE USED TO NORMALIZE DATA
          IORM = ITFLAG(NR)
          IF (JNDRM = 1) I00,110,120

          C
          C NORMALIZE W.R.T. A,R
          100 IX(NR) = 4
          1Y(NR) = 5
          GO TO 190

          C
          C NORMALIZE W.R.T. X,Y

```

FLNT 176
 FLNT 177
 FLNT 178
 FLNT 179
 FLNT 180
 FLNT 181
 FLNT 182
 FLNT 183
 FLNT 184
 FLNT 185
 FLNT 186
 FLNT 187
 FLNT 188
 FLNT 189
 FLNT 190
 FLNT 191
 FLNT 192
 FLNT 193
 FLNT 194
 FLNT 195
 FLNT 196
 FLNT 197
 FLNT 198
 FLNT 199
 FLNT 200
 FLNT 201
 FLNT 202
 FLNT 203
 FLNT 204
 FLNT 205
 FLNT 206
 FLNT 207
 FLNT 208
 FLNT 209
 FLNT 210

FLNTRP

FLNTRP

FLNT	211
FLNT	212
FLNT	213
FLNT	214
FLNT	215
FLNT	216
FLNT	217
FLNT	218
FLNT	219
FLNT	220
FLNT	221
FLNT	222
FLNT	223
FLNT	224
FLNT	225
FLNT	226
FLNT	227
FLNT	228
FLNT	229
FLNT	230
FLNT	231
FLNT	232
FLNT	233
FLNT	234
FLNT	235
FLNT	236
FLNT	237
FLNT	238
FLNT	239
FLNT	.00
FLNT	241
FLNT	242
FLNT	243
FLNT	244
FLNT	245

FLNTRP

```

110 IX(NR) = 1
    IY(NR) = 2
    GO TO 190
C
120 IF (INORM = 3) 130,140,100
C  NORMALIZE W,R,T  X,Z
130 IX(NR) = 1
    IY(NR) = 3
    GO TO 190
C
C  NORMALIZE W,R,T.  Y,Z
140 IX(NR) = 2
    IY(NR) = 3
    GO TO 190
C
C
C
C  BRING THE FLOW DATA INTO CORE
C  START WITH BOUNDARY ONE
190 IC = 0
    IB = 0
    MF = 1
    LS = 2
    IXX = IX(NR)
    IYY = IY(NR)
    A1(NR) = 0.0
    A2(NR) = 0.0
    DO 195 J = 1,6
195 FLOWC(NR,J) = 0.0
    GO TO 205
200 A1(NR) = XI(1)
    A2(NR) = XI(NB1)
205 IA = IB + 1
    NB1 = IFC(NR,IA)
    NB3 = NB1 + 3

```

FLNTRP

FLNTR 246
 FLNTR 247
 FLNTR 248
 FLNTR 249
 FLNTR 250
 FLNTR 251C
 FLNTR 252I
 FLNTR 253
 FLNTR 254
 FLNTR 255
 FLNTR 256
 FLNTR 257
 FLNTR 258
 FLNTR 259
 FLNTR 260
 FLNTR 261
 FLNTR 262
 FLNTR 263
 FLNTR 264
 FLNTR 265
 FLNTR 266
 FLNTR 267
 FLNTR 268
 FLNTR 269
 FLNTR 270
 FLNTR 271
 FLNTR 272
 FLNTR 273
 FLNTR 274
 FLNTR 275
 FLNTR 276
 FLNTR 277
 FLNTR 278
 FLNTR 279
 FLNTR 280

```

    IYAG10 = LOFF(IB)
    FIC(NR,IB) = 0.0
  C
    DO 210 I = 1,NB1
      IC = IC + 1
      CALL READMS (10,E4,12,IYAG10)
      READ (10,IYAG10) E4
      IYAG10 = IYAG10 + 1
      DO 207 II=1,6
        207 COORDS(II) = E4(II)
      DO 208 II=1,6
        208 FLOW(IC,II) = E4(II+6)
      XI(I) = COORDS(IXX)
      YI(I) = 1.0
      FI(I) = COORDS(IYY)
      FIC(NR,IB) = FIC(NR,IB) + FI(I)
      DO 206 J = 1,6
        206 FLOWC(NR,J) = FLOWC(NR,J) + FLOW(IC,J)
      XB(NR,IC) = COORDS(IXX)
      XX(NR,IC) = COORDS(IXX)
      YY(NR,IC) = COORDS(IYY)
      210 CONTINUE
  C
      WRITE(TAPE0, 3050) IB, (XI(I), YI(I), FI(I), I = 1,NB1)
      3050 FORMAT(IH0, 5HIB = ,I2/, (1H ,F20.6))
  C CHECK IF COEFFICIENTS ALREADY AVAILABLE
      IF (LOCD(IB) .NE. 0) GO TO 250
      FIC(NR,IB) = FIC(NR,IB)/NB1
      DO 215 I = 1,NB1
        215 FI(I) = FI(I) * FIC(NR,IB)
  C CALCULATE BOUNDARY COEFFICIENTS
  C CALL ROWFMI(NB1,MF,XI,YI,FI,XKF,MX,MX1,LS, TAPEB)
  C CALL SOLVIT(A,N3,MF,KD, TAPEB,TAPEC,TAPEB, B,MX,MX, NERR)

```

FLNTRP

FLNTRP

```

C   C   CHECK IF MATRIX SINGULAR
      IF (NERR .EQ. 0) GO TO 230
      WRITE(6, 220)
      220 FORMAT('M1, 16MMATRIX SINGULAR. ')
C
C   C   PUT COEFFICIENTS INTO PROPER ARRAY AND STORE
      230 CONTINUE
      DO 240 I = 1, N3
      240 CFR(NR, IB, I) = R(I, I)
C
      ITAG10 = IMTAB(2)
      LOCD(IB) = ITAG10
      I6 = N3 + 3
      IFD(NR, IB) = I6
      I2 = 0
      241 I1 = I2 + 1
      I2 = I2 + 25
      IF (I2 .GT. N3) I2 = N3
      II = 0
      DO 242 I = I1, I2
      II = II + 1
      242 FS(II) = B(I, I)
C
      CALL WRITMS(10, ES, 25, ITAG10)
C
C   C   WRITE(10, ITAG10) ES
      ITAG10 = ITAG10 + 1
      IF (I2 .LT. N3) GO TO 241
      E7(1) = A1(NR)
      E7(2) = A2(NR)
      E7(3) = FIC(NR, IB)
C
      CALL WRITMS(10, E7, 3, ITAG10)

```

FLNTRP

FLNTRP

```

C
C   WRITE(10,ITAG10) E7
C   IMTAB(2) = ITAG10 + 1
C   WRITE(TAPEOT, 3055) IB, (B(I,1),I=1,N3), A1(NR),A2(NR),FIC(NR,IB)
C   3055 FORMAT(1H0, 8BOUNDARY,I2,13H COEFFICIENTS//((1H , 7F18.6))
C
C   GO TO 300
C
C   READ YN BOUNDARY ONE COEFFICIENTS
C   250 ITAG10 = LOCD(IB)
C   I6 = N3 + 3
C   I2 = 0
C   251 I1 = I2 + 1
C   I2 = I2 + 25
C   IF (I2 .GT. N ) I2 = N3
C
C   CALL READMS(,E5,25,ITAG10)
C
C   READ(10,ITAG1 E5
C   ITA = 0 = ITAG1.0 + 1
C   II = 0
C   DO 252 I = I1,I2
C   II = II + 1
C   252 CFR(NR,IB,I) = E5(II)
C   IF (I2 .LT. N3) GO TO 251
C
C   CALL READMS(10,E7,3,ITAG10)
C
C   READ(10,ITAG10) E7
C   A1(NR) = E7(1)
C   A2(NR) = E7(2)
C   FIC(NR,IB) = E7(3)
C
C   NOW DO BOUNDARY TWO

```

```

FLMT 316
FLMT 317I
FLMT 318
FLMT 319
FLMT 320
FLMT 321
FLMT 322
FLMT 323
FLMT 324
FLMT 325
FLMT 326
FLMT 327
FLMT 328
FLMT 329
FLMT 330
FLMT 331
FLMT 332C
FLMT 333
FLMT 334I
FLMT 335
FLMT 336
FLMT 337
FLMT 338
FLMT 339
FLMT 340
FLMT 341
FLMT 342C
FLMT 343
FLMT 344I
FLMT 345
FLMT 346
FLMT 347
FLMT 348
FLMT 349
FLMT 350

```

FLNTRP

FLNTRP

FLNT 351
 FLNT 352
 FLNT 353
 FLNT 354
 FLNT 355
 FLNT 356
 FLNT 357
 FLNT 358
 FLNT 359C
 FLNT 360I
 FLNT 361
 FLNT 362
 FLNT 363
 FLNT 364
 FLNT 365
 FLNT 366
 FLNT 367
 FLNT 368
 FLNT 369
 FLNT 370
 FLNT 371
 FLNT 372
 FLNT 373
 FLNT 374
 FLNT 375
 FLNT 376
 FLNT 377
 FLNT 378
 FLNT 379
 FLNT 380
 FLNT 381
 FLNT 382
 FLNT 383
 FLNT 384
 FLNT 385

```

300 IF (IB .EQ. 1) GO TO 200
C BOTH BOUNDARIES COMPLETE. READ ADDITIONAL DATA (IF ANY).
  NB1 = IFC(NR,5)
  IF (NB1 .EQ. 0) GO TO 320
  ITAG10 = LOFF(S)
  DO 310 I = 1, NB1
    IC = IC + 1
    CALL READMS (10, E4, 12, ITAG10)
    READ (10, ITAG10) E4
    ITAG10 = ITAG10 + 1
    DO 301 J = 1, 6
      301 COORDS(J) = E4(J)
    DO 302 J = 1, 6
      302 FLOW(IC, J) = E4(J+6)
    XX(NR, IC) = COORDS(1XX)
    YY(NR, IC) = COORDS(1YY)
    DO 305 J = 1, 6
      305 FLOWC(NR, J) = FLOWC(NR, J) + FLOW(IC, J)
  310 CONTINUE
C NORMALIZE DATA POINTS
  320 IFC(NR, 6) = IC
    AX = 1.0 / (A2(NR) * A1(NR))
C **NOTE. BOUNDARY ONE POINTS HAVE RN = 0.0
C BOUNDARY TWO POINTS HAVE RN = 1.0
C
  NB1 = IFC(NR, 1)
  DO 330 I = 1, NB1
    XI(I) = (XX(NR, I) * A1(NR)) * AX
    330 YI(I) = 0.0
    NI = NB1 + 1
    NB1 = NB1 + IFC(NR, 2)
    DO 340 I = NI, NB1

```

FLNTRP

FLNTRP

FLMT 386
 FLMT 387
 FLMT 388
 FLMT 389
 FLMT 390
 FLMT 391
 FLMT 392
 FLMT 393
 FLMT 394
 FLMT 395
 FLMT 396
 FLMT 397
 FLMT 398
 FLMT 399
 FLMT 400
 FLMT 401
 FLMT 402
 FLMT 403
 FLMT 404
 FLMT 405
 FLMT 406
 FLMT 407
 FLMT 408
 FLMT 409
 FLMT 410
 FLMT 411
 FLMT 412
 FLMT 413
 FLMT 414
 FLMT 415
 FLMT 416
 FLMT 417
 FLMT 418
 FLMT 419
 FLMT 420

```

    C
    C 340 XI(I) = (XX(NR,I) - A1(NR))*AX
    C      YI(I) = 1.0
    C
    C      N2 = IFC(NR,6)
    C      IF (IFC(NR,5) .EQ. 0) GO TO 501
    C
    C FIELD POINTS
    C N1 = N2 + IFC(NR,5) + 1
    C DO 500 I = N1,N2
    C   IB = 0
    C   IC = 0
    C
    C CALCULATE BOTH BOUNDARY VALUES
    C 410 IB = IB + 1
    C      N = IFC(NR,IR)
    C      N3 = N + 3
    C      RB(IB) = CFB(NR,IB,1) + CFB(NR,IB,2)*YY(NR,I) + PIC(NR,IB)
    C      J3 = 3
    C      DO 450 J = 1,N
    C        J3 = J3 + 1
    C        UIJ = 0.0
    C        IC = IC + 1
    C        IF (XX(NR,I) .EQ. XX(NR,IC)) GO TO 430
    C        TR1 = (XX(NR,I) - XX(NR,IC))*2
    C        UIJ = TRI*ALOG(TR1)
    C 430 RB(IR) = RB(IB) + UIJ*CFB(NR,IB,J3)
    C 450 CONTINUE
    C      IF (IB .EQ. 1) GO TO 410
    C
    C BOTH BOUNDARIES CALCULATE, NORMALIZE POINT
    C XI(I) = (XX(NR,I) - A1(NR))*AX
    C YI(I) = (YY(NR,I) - RB(1))/(RB(2) - RB(1))
    C 500 CONTINUE
    C
    C 501 WRITE(TAPEOT, 3060) NR
  
```

FLNTRP

FLNTRP

```

3060 FORMAT(1H0,36HNORMALIZED FLOW FIELD DATA REGION = ,I3, 10X,
1 21HXI, YI, (FI(J),J=1,6)//)
WRITE(TAPEOT, 3070) (XI(I), YI(I), (FLOW(I,J),J=1,6), I=1,N2)
3070 FORMAT(1H , 8F16,6)
C RELOAD COORDINATE ARRAYS
502 DO 503 I = 1,N2
YY(NR,I) = YI(I)
503 XX(NR,I) = XI(I)
C
C CHECK IF FLOW FIELD COEFFICIENTS AVAILABLE
N3 = N2 + 3
IF (LOCDS) .NE. 0) GO TO 550
DO 505 J = 1,6
FLOWC(NR,J) = FLOWC(NR,J)/N2
DO 504 I = 1,N2
504 FLOW(I,J) = FLOW(I,J) * FLOWC(NR,J)
505 CONTINUE
C
C CALCULATE COEFFICIENTS FOR FLOW FIELD DATA
NB1 = N2
LS = 0
CALL ROWFMI(NB1,MX,XI,YI,FLOW,XKF,NX,MX,LS, TAPEB)
C
CALL SOLVIT(A,N3,MX,KD, TAPEB,TAPEC,TAPED,0,NX,MX, NERR)
C
IF (NERR .NE. 0) WRITE(6, 220)
C
C PUT COEFFICIENTS IN PROPER ARRAYS AND STORE
DO 520 I = 1,N3
DO 510 J = 1,6
510 CFF(NR,I,J) = B(I,J)
520 CONTINUE
C
ITAG10 = IMTAB(2)
LOCDS(5) = ITAG10

```

FLNT 421
FLNT 422
FLNT 423
FLNT 424
FLNT 425
FLNT 426
FLNT 427
FLNT 428
FLNT 429
FLNT 430
FLNT 431
FLNT 432
FLNT 433
FLNT 434
FLNT 435
FLNT 436
FLNT 437
FLNT 438
FLNT 439
FLNT 440
FLNT 441
FLNT 442
FLNT 443
FLNT 444
FLNT 445
FLNT 446
FLNT 447
FLNT 448
FLNT 449
FLNT 450
FLNT 451
FLNT 452
FLNT 453
FLNT 454
FLNT 455

FLNTRP

FLNTRP

456 FLNT
 457 FLNT
 458 FLNT
 459 FLNT
 460 FLNT
 461 FLNT
 462 FLNT
 463 FLNT
 464 FLNT
 465 FLNT
 466 FLNT
 467 FLNT
 468C FLNT
 469 FLNT
 470I FLNT
 471 FLNT
 472 FLNT
 473 FLNT
 474 FLNT
 475 FLNT
 476C FLNT
 477 FLNT
 478I FLNT
 479 FLNT
 480 FLNT
 481 FLNT
 482 FLNT
 483 FLNT
 484 FLNT
 485 FLNT
 486 FLNT
 487 FLNT
 488C FLNT
 489I FLNT
 490 FLNT

```

I6 = N3*6
I2 = 0
521 I1 = I2 + 1
    I2 = I2 + 4
    IF (I2 .GT. N3) I2 = N3
    II = 0
    DO 523 I = I1, I2
      DO 522 J = 1, 6
        II = II + 1
522 E5(II) = 8(I, J)
523 CONTINUE
C
C CALL WRITHS(10, E5, 24, ITAG10)
C
C WRITE(10, ITAG10) E5
ITAG10 = ITAG10 + 1
IF (I2 .LT. N3) GO TO 521
DO 524 J = 1, 6
524 E6(J) = FLOWC(NR, J)
C
C CALL WRITHS(10, E6, 6, ITAG10)
C
C WRITE(10, ITAG10) E6
IPD(NR, 5) = I6 + 6
IMTAR(2) = ITAG10 + 1
WRITE(TAPEOT, 3065) ((B(I, J), J=1, 6), I=1, N3), (FLOWC(NR, J), J=1, 6)
3065 FORMAT(1X0, 17#FLOW COEFFICIENTS/(1H, 6P18.6))
C
C RESET DIRECTORIES
DO 526 I = 1, 5
  E1(I+10) = LOCD(I)
526 E1(I+15) = IPD(NR, I)
CALL WRITHS(10, E1, 25, IG10)
C
C WRITE(10, IG10) E1
ITAG10 = 2

```

FLNTRP

FLNTRP

```

C      CALL WRITMS(10, IMTAB, 9, ITAG10)
C      WRITE(10,ITAG10) IMTAB
C      GO TO 600

C      READ IN FLOW FIELD COEFFICIENTS
C      550 ITAG10 = LOLO(5)
C      I6 = N3+6
C      I2 = 0
C      551 I1 = I2 + 1
C      I2 = I2 + 4
C      IF (I2 .GT. N3) I2 = N3

C      CALL READMS(10,E5,E24,ITAG10)

C      READ(10,ITAG10) E5
C      ITAG10 = ITAG10 + 1
C      I1 = 0
C      DO 553 I = I1,I2
C      DO 552 J = 1,6
C      I1 = I1 + 1
C      552 CFF(NR,I,J) = E5(I1)
C      553 CONTINUE
C      IF (I2 .LT. N3) GO TO 551

C      CALL READMS(10,E6,E6,ITAG10)

C      READ(10,ITAG10) E6
C      DO 554 J = 1,6
C      554 FLOWC(NR,J) = E6(J)

C      600 CONTINUE

C      COST(NR) = COS(THETO(NR)*RC)

```

```

FLNT 491C
FLNT 492I
FLNT 493
FLNT 494
FLNT 495
FLNT 496
FLNT 497
FLNT 498
FLNT 499
FLNT 500
FLNT 501
FLNT 502
FLNT 503C
FLNT 504
FLNT 505I
FLNT 506
FLNT 507
FLNT 508
FLNT 509
FLNT 510
FLNT 511
FLNT 512
FLNT 513
FLNT 514
FLNT 515C
FLNT 516
FLNT 517I
FLNT 518
FLNT 519
FLNT 520
FLNT 521
FLNT 522
FLNT 523
FLNT 524
FLNT 525

```

FLNTRP

FLMT 526
 FLMT 527
 FLMT 528
 FLMT 529
 FLMT 530
 FLMT 531
 FLMT 532
 FLMT 533
 FLMT 534
 FLMT 535
 FLMT 536
 FLMT 537
 FLMT 538
 FLMT 539
 FLMT 540
 FLMT 541
 FLMT 542
 FLMT 543
 FLMT 544
 FLMT 545
 FLMT 546
 FLMT 547
 FLMT 548
 FLMT 549
 FLMT 550
 FLMT 551
 FLMT 552
 FLMT 553
 FLMT 554
 FLMT 555
 FLMT 556
 FLMT 557
 FLMT 558
 FLMT 559
 FLMT 560

```

SINT(NR) = SIN(THETO(NR)*RC)
COSS(NR) = COS(PSIO(NR)*RC)
SINS(NR) = SIN(PSIO(NR)*RC)
1000 CONTINUE
C COEFFICIENTS FOR ALL REQUESTED REGIONS ARE IN CORE
C START CYCLE ON ELEMENT DATA
1010 CONTINUE
WRITE(TAPEUT, 3080)
3080 FORMAT(1H0,4MINTERPOLATED DATA, NR, XP(1XX), YP(1YY), AN, RN,,
117H(DINFL(1), I=1,6)//)
WRITE(TAPEUT, 3082) LTOT
3082 FORMAT(1H , 7HLTOT = , IS//)
ISYM = 1
ICT = 0
REWIND TAPE
1011 DO 1500 II = 1,LTOT
READ (TAPEE,END=1510) IG4,ELEM
READ (TAPEE) IG4,ELEM
IF (EOF(TAPEF)) 1510,1012
1012 LCOND(2) = ELEM(1)
IF (ISYM .EQ. 1) GO TO 1090
IG4 = *IG4
C REFLECT ELEMENT TO *Y SIDE AND CHANGE ORDER OF PTS 2 AND 4
ELEM(5) = *ELEM(5)
ELEM(8) = *ELEM(8)
ELEM(15) = *ELEM(15)
ELEM(16) = *ELEM(16)
ELEM(17) = *ELEM(17)
ELEM(18) = *ELEM(18)
ELT = ELEM(12)
ELEM(12) = ELEM(14)
ELEM(14) = ELT
ELT = ELEM(16)
ELEM(16) = ELEM(18)
  
```

FLNTRP

FLNT 561
 FLNT 562
 FLNT 563
 FLNT 564
 FLNT 565
 FLNT 566
 FLNT 567
 FLNT 568
 FLNT 569
 FLNT 570
 FLNT 571
 FLNT 572
 FLNT 573
 FLNT 574
 FLNT 575
 FLNT 576
 FLNT 577
 FLNT 578
 FLNT 579
 FLNT 580
 FLNT 581
 FLNT 582
 FLNT 583
 FLNT 584
 FLNT 585
 FLNT 586
 FLNT 587
 FLNT 588
 FLNT 589
 FLNT 590
 FLNT 591
 FLNT 592
 FLNT 593
 FLNT 594
 FLNT 595

FLNTRP

```

ELEM(18) = ELT
ELT = ELEM(20)
FLFM(20) = ELEM(22)
FLFM(22) = ELT
1090 IF (NFR .EQ. 0) GO TO 1155
XP(1) = ELEM(7)
XP(2) = ELEM(8)
XP(3) = ELEM(9)
C SFT FLOW FIELD REGION NUMBER
N = 1
C
C TRANSFORM COORDS TO FLOW PLANE ORIENTATION
1100 CONTINUE
AA11 = COST(NR)*CCSS(NR)
AA12 = COST(NP)*SINS(NR)
AA13 = *SINT(NR)
AA21 = -SINS(NR)
AA22 = CCSS(NR)
AA23 = 0.0
AA31 = -SINT(NR)*CCSS(NR)
AA32 = -SINT(NR)*SINS(NR)
AA33 = -COST(NR)
DX = XP(1) * XD(NR)
DY = XP(2) * YD(NR)
DZ = XP(3) * ZD(NR)
C AXIAL COORD
XP(4) = AA11*DX + AA12*DY + AA13*DZ
C
C RADIAL COORD
YP = AA21*DX + AA22*DY
ZP = AA31*DX + AA32*DY + AA33*DZ
XP(5) = SQRT(YP**2 + ZP**2)
C ANGULAR COORD
IF (YP .NE. 0.0) GO TO 1102
PHT = 0.0
  
```

FLNTRP

FLNT 599
 FLNT 597
 FLNT 598
 FLNT 599
 FLNT 600
 FLNT 601
 FLNT 602
 FLNT 603
 FLNT 604
 FLNT 605
 FLNT 606
 FLNT 607
 FLNT 608
 FLNT 609
 FLNT 610
 FLNT 611
 FLNT 612
 FLNT 613
 FLNT 614
 FLNT 615
 FLNT 616
 FLNT 617
 FLNT 618
 FLNT 619
 FLNT 620
 FLNT 621
 FLNT 622
 FLNT 623
 FLNT 624
 FLNT 625
 FLNT 626
 FLNT 627
 FLNT 628
 FLNT 629
 FLNT 630

FLNTRP

```

IF (ZP .EQ. 0.0) GO TO 1103
1102 PHI = ATAN2(YP,ZP)
1103 XP(6) = PHI * PHIO(NR)
SINP = SIN(PHI)
COSP = COS(PHI)

```

C

```

AP11 = AA11
AP12 = AA12
AP13 = AA13
AP21 = COSP*AA21 - SINP*AA31
AP22 = COSP*AA22 + SINP*AA32
AP23 = -SINP*AA33
AP31 = SINP*AA21 + COSP*AA31
AP32 = SINP*AA22 + COSP*AA32
AP33 = COSP*AA33

```

C NORMALIZE COORDS FOR FLOW REGION, FIRST FIND BOUNDARIES.

```

IXX = IX(NR)
IYY = IY(NR)
IB = 0
IC = 0

```

```

1110 IB = IB + 1
N1 = IFC(NP,IB)
N3 = N1 + 3
RR(IB) = CFB(NR,IB,1) + CF9(NR,IB,2)*XP(IXX) + FIC(NR,IB)
J3 = 3

```

C

```

DO 1150 J = 1,N1
J3 = J3 + 1
IC = IC + 1
UIJ = 0.0
IF (XP(IXX) .EQ. XB(NR,IC)) GO TO 1130
TRI = (XP(IXX) - XB(NR,IC))*2
UIJ = TRI*ALOG(TRI)
1130 RR(IB) = RB(IB) + UIJ*CFB(NR,IB,J3)
1150 CONTINUE

```

FLNTRP

```

C IF (IB .EQ. 1) GO TO 1110
C BOTH BOUNDARIES CALCULATED, FORM NORMALIZED COORDS.
  AN = (XP(IXX) - A1(NR))/(A2(NR) - A1(NR))
  RN = (XP(IYY) - RB(1))/(RB(2) - RB(1))
  WRITE(TAPE07, 3083) RB
3083 FORMAT(1H, 8HRB(1) = , F18.6, 10X, 8HRB(2) = , F18.6)
C TEST IF NORMALIZED COORDS WITHIN REGION
  IF ((AN .LT. 0.0)) GO TO 1152
  IF ((RN .GE. 0.0) .AND. (RN .LE. 1.0)) GO TO 1200
C POINT NOT WITHIN REGION, GO TO NEXT REGION (IF SPECIFIED),
  1152 CONTINUE
  NR = NR + 1
  IF (NR .LE. NFR) GO TO 1100
C NOT WITHIN ANY SPECIFIED REGION. DEFAULT TO FREE STREAM.
  1155 LCONVD(1) = 0
  LCONVD(3) = 0
  DO 1160 K = 1,6
  1160 DINF(LK) = 0.0
  GO TO 1400
C INTERPOLATE FOR FLOW FIELD VARIABLES
  1200 CONTINUE
  N1 = IFC(NR,6)
  N3 = N1 + 3
  XKF(1) = 1.0
  XKF(2) = AN
  XKF(3) = RN
  J3 = 3
C DO 1250 J = 1,N1
  J3 = J3 + 1
  UIJ = 0.0
```

631 FLNT
632 FLNT
633 FLNT
634 FLNT
635 FLNT
636 FLNT
637 FLNT
638 FLNT
639 FLNT
640 FLNT
641 FLNT
642 FLNT
643 FLNT
644 FLNT
645 FLNT
646 FLNT
647 FLNT
648 FLNT
649 FLNT
650 FLNT
651 FLNT
652 FLNT
653 FLNT
654 FLNT
655 FLNT
656 FLNT
657 FLNT
658 FLNT
659 FLNT
660 FLNT
661 FLNT
662 FLNT
663 FLNT
664 FLNT
665 FLNT

FLNTRP

FLNTRP

```

        TR1 = 0.0
        TR2 = 0.0
        IF (AN.EQ. XX(NR,J)) GO TO 1210
        TR1 = (AN * XX(NR,J))**2
1210  IF (RN.EQ. YY(NR,J)) GO TO 1220
        TR2 = (RN * YY(NR,J))**2
1220  RBIJ = TR1 + TR2
        IF (RBIJ.EQ. 0.0) GO TO 1230
        UIJ = RBIJ*ALOG(RBIJ)
1230  XKF(J3) = UIJ
1250  CONTINUE
C
C  CALCULATE FUNCTIONS
        DO 1300 K = 1,6
        DINF(K) = FLOWC(NR,K)
        DO 1290 J = 1,N3
        DINF(K) = DINF(K) + XKF(J)*CFF(NR,J,K)
1290  CONTINUE
1300  CONTINUE
C  TRANSFORM DIRECTION COSINES
        VX = DINF(2)
        VY = DINF(3)
        VZ = DINF(4)
        VT = SQRT(VX**2 + VY**2 + VZ**2)
        DINF(2) = VX(0.0)/VT
        DINF(3) = VY(0.0)/VT
        DINF(4) = VZ(0.0)/VT
C
C  SAVE RESULTS
        LCOND(1) = 1
        LCOND(3) = NR
        WRITE(TAPEOT,3084) VX,VY,VZ
3084  FORMAT(6H VX = ,F12.6,10H
1400  CONTINUE
        WRITE(TAPEOT, 3085) LCOND, (ELEM(I), I=4,10)
        VY = ,F12.6, 10H
        VZ = ,F12.6)

```

FLNT 666
 FLNT 667
 FLNT 668
 FLNT 669
 FLNT 670
 FLNT 671
 FLNT 672
 FLNT 673
 FLNT 674
 FLNT 675
 FLNT 676
 FLNT 677
 FLNT 678
 FLNT 679
 FLNT 680
 FLNT 681
 FLNT 682
 FLNT 683
 FLNT 684
 FLNT 685
 FLNT 686
 FLNT 687
 FLNT 688
 FLNT 689
 FLNT 690
 FLNT 691
 FLNT 692
 FLNT 693
 FLNT 694
 FLNT 695
 FLNT 696
 FLNT 697
 FLNT 698
 FLNT 699
 FLNT 700

FLNTRP

FLNTRP

FLNT 701
 FLNT 702
 FLNT 703
 FLNT 704
 FLNT 705
 FLNT 706
 FLNT 707
 FLNT 708
 FLNT 709
 FLNT 710
 FLNT 711
 FLNT 712
 FLNT 713
 FLNT 714
 FLNT 715
 FLNT 716
 FLNT 717
 FLNT 718
 FLNT 719
 FLNT 720
 FLNT 721
 FLNT 722
 FLNT 723
 FLNT 724
 FLNT 725
 FLNT 726
 FLNT 727C
 FLNT 728I
 FLNT 729
 FLNT 730
 FLNT 731
 FLNT 732
 FLNT 733
 FLNT 734
 FLNT 735

```

3085 FORMAT(1M, 3I5, 2X, 7F14.6)
WRITE(TAPEOT, 3090) NR, XP(IXX), XP(IYY), AN, RN,(DINFL(I),I=1,6)
3090 FORMAT(1M, I3, 10F12.6)
IF (LCOND(3) .NE. 0) GO TO 3100
IFLNR = 0
ISRNR = 0
ISFNR = 0
GO TO 3110
3100 IFLNR = IFL(NR)
ISRNR = ISR(NR)
ISFNR = ISF(NR)
3110 CONTINUE
WRITE(TAPEF) IG4,LCOND,ELEM,DINFL,NDSET,PARSET,IFLNR,ISRNR,ISFNR
ICT = ICT + 1
C GO TO NEXT ELEMENT
1500 CONTINUE
1510 CONTINUE
C CHECK FOR SYMMETRY REQUIREMENTS
IF (SYMECT.EQ.1 .OR. BETAS.EQ.0.0 .OR. ISYM.EQ.2) GO TO 2010
REWIND TAPEE
ISYM = 2
GO TO 1011
C UNIFORM FLOW FIELD IS ON DATA STORAGE UNIT. LOAD INTO DIMF ARRAY,
1600 ITAG10 = LOG(IR) + 5
CALL READMS (10,DINFL,6,ITAG10)
C READ (10=ITAG10) DINFL
DO 1610 I=1,6
1610 DIMF(IAB,I) = DINFL(I)
GO TO 2000
C
2010 LCTAR(IAB) = ICT
C THIS MEANS THAT TAPEF MUST BE FILLED ON THE SAME RUN THAT THE
C FORCES ARE CALCULATED SINCE THERE IS NO OTHER WAY TO FILL UP THE

```

FLNTRP

FLNTRP

```
C LYOTAP ARRAY WITH THE PROPER ELEMENT COUNTS.  
C END OF ALPHA-BETA LOOP  
C 2600 CONTINUE  
C REWIND TAPEE  
C REWIND TAPEF  
C RETURN  
C END
```

```
FLNT 736  
FLNT 737  
FLNT 738  
FLNT 739  
FLNT 740  
FLNT 741  
FLNT 742  
FLNT 743  
FLNT 744
```

FLNTRP

VAL2E

```

SURROUTINE VALU2(N,X,Y,Z,XN,YN,ZN,IC,IT,INT,EMT,EMX,EMY,EMZ,POP,
1 TOT)
C
COMMON/SURFH2/INORM,ISURF,IX,IY,IZ,U1,U2,V1,VL,CR,CT,CHX,N2,N3,
1 XO,YO,ZO,AP11,AP12,AP13,AP21,AP22,AP23,AP31,AP32,AP33,
2 FLOWC(7),XI(500),YI(500),B(503,7),XKF(503),AA(5999),
A XR(4),YB(4),ZB(4),
3 N0SET,IABSET,IR,LORG(20),ISR(20),NS,ISFR,IVIS,IFTYP,IFLOW,
4 BY(2,20),BY(2,20),NB
C
DIMENSION X(N),Y(N),Z(N),INT(N),EMT(N),EMX(N),EMY(N),EMZ(N),
1 POP(N),TOT(N),VT(6),XT(6),FI(7),BF(2)
C
C
DO 500 I = 1,M
IF (IC .EQ. 1) GO TO 70
C
TRANSFORM COORDS
XX = X(I) - XO
YY = Y(I) - YO
ZZ = Z(I) - ZO
C
XT(1) = XX*AP11 + YY*AP12 + ZZ*AP13
XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
XT(4) = XT(1)
XT(5) = SQRT(XT(2)**2 + XT(3)**2)
IF (XT(2) .EQ. 0.0) GO TO 40
XT(6) = ATAN2(XT(2),-XT(3))
GO TO 50
40 XT(6) = 0.0
IF (XT(3) .GT. 0.0) XT(6) = 3.141592666
50 CONTINUE
C
NORMALIZE COORDS

```

VAL2E

VALZE

VAL2 036
VAL2 038
VAL2 039
VAL2 040
VAL2 041
VAL2 042
VAL2 043
VAL2 044
VAL2 045
VAL2 046
VAL2 047
VAL2 048
VAL2 049
VAL2 050
VAL2 051
VAL2 052
VAL2 053
VAL2 054
VAL2 055
VAL2 056
VAL2 057
VAL2 058
VAL2 059
VAL2 060
VAL2 061
VAL2 062
VAL2 063
VAL2 064
VAL2 065
VAL2 066
VAL2 067
VAL2 068
VAL2 069
VAL2 070

```
YN = (X*(IY) - V1)/VL
IF (ISURF .EQ. 1) GO TO 60
XN = (XT(IX) - U1)/UL
GO TO 70
60 CHX = CR + (CT-CR)*YN
XLE = U1 + (UL-U1)*YN
XN = (XT(IX) - XLE)/CHX

C CHECK IF WITHIN SURFACE BOUNDARIES
70 IF (XN .LT. 0.0) GO TO 80
IF (XN .GT. 1.0) GO TO 80
IF (YN .LT. 0.0) GO TO 80
IF (YN .GT. 1.0) GO TO 80

C CHECK IF WITHIN FLOW BOUNDARIES
IB = 0
800 IB = IB + 1
DO 810 I1 = 1,NS
  J = I1
  IF (YN .LT. BY(IB,I1)) GO TO 820
810 CONTINUE

C
820 IF (J .EQ. 1) J = 2
DY = BY(IB,J) - BY(IB,J-1)
IF (DY .EQ. 0.0) GO TO 80
DX = BX(IB,J) - BX(IB,J-1)
RP(IR) = BX(IB,J-1) + (YN - BY(IR,J-1))*DX/DY
GO TO (830,830), IB
830 IF (XN .LT. BP(IR)) GO TO 80
GO TO 800
840 IF (XN .GT. RP(IR)) GO TO 80

C POINT WITHIN FLOW BOUNDARIES
XN = XN + BP(I)
IF (IVIS .EQ. 0) GO TO 850
```

VALZE

VAL2E

```

      XN = XN/(BX(2,1) * BX(1,1))
      GO TO 860
      850 XN = XN/(BP(2) * BP(1))
      860 CONTINUE

```

C POINT WITHIN BOUNDARIES, CALCULATE VALUES,

```

      XKF(1) = 1.0
      XKF(2) = XN
      XKF(3) = YN
      J3 = 3

```

C DO 250 J = 1,N2

```

      J3 = J3 + 1
      UIJ = 0.0
      TR1 = 0.0
      TR2 = 0.0
      IF (XN.EQ. XI(J)) GO TO 210
      TR1 = (XN - XI(J))*2
      210 IF (YN.EG. YI(J)) GO TO 220
      TR2 = (YN - YI(J))*2
      220 RBIJ = TR1 + TR2
      IF (RBIJ.EQ. 0.0) GO TO 230
      UIJ = RBIJ*ALOG(RBIJ)
      230 XKF(J3) = UIJ
      250 CONTINUE

```

C

```

      DO 300 K = 1,7
      FI(K) = FLOWC(K)
      DO 290 J = 1,N3
      FI(K) = FI(K) + XKF(J)*B(J,K)
      290 CONTINUE
      300 CONTINUE

```

C INT(I) = 1

VAL2 071
 VAL2 072
 VAL2 073
 VAL2 074
 VAL2 075
 VAL2 076
 VAL2 077
 VAL2 078
 VAL2 079
 VAL2 080
 VAL2 081
 VAL2 082
 VAL2 083
 VAL2 084
 VAL2 085
 VAL2 086
 VAL2 087
 VAL2 088
 VAL2 089
 VAL2 090
 VAL2 091
 VAL2 092
 VAL2 093
 VAL2 094
 VAL2 095
 VAL2 096
 VAL2 097
 VAL2 098
 VAL2 099
 VAL2 100
 VAL2 101
 VAL2 102
 VAL2 103
 VAL2 104
 VAL2 105

VAL2E

VAL2E

VAL2 106
VAL2 107
VAL2 108
VAL2 109
VAL2 110
VAL2 111
VAL2 112
VAL2 113
VAL2 114
VAL2 115
VAL2 116
VAL2 117
VAL2 118
VAL2 119
VAL2 120
VAL2 121
VAL2 122
VAL2 123
VAL2 124
VAL2 125
VAL2 126
VAL2 127
VAL2 128
VAL2 129
VAL2 130
VAL2 131
VAL2 132
VAL2 133
VAL2 134
VAL2 135
VAL2 136
VAL2 137
VAL2 138
VAL2 139
VAL2 140

```
EMT(I) = FI(1)
FT = SQRT(FI(2)**2 + FI(3)**2 + FI(4)**2)
IF (FT .EQ. 0.0) FT = 1.0
FI(2) = FI(2)/FT
FI(3) = FI(3)/FT
FI(4) = FI(4)/FT
EMX(I) = FI(2)/CR
EMY(I) = FI(3)/VL
POP(I) = FI(5)
TOT(I) = FI(6)
ZN = FI(7)

C TRANSFORM XM, YM, ZN COORDS BACK TO (X, Y, Z)
IF (IVIS .EQ. 0) GO TO 310
XN = XN*(BX(2,1) + BX(1,1))
GO TO 320
310 XN = XN*(BP(2) + BP(1))
320 XN = XN + BP(1)
XT(IY) = YN*VL + VI
YT(IZ) = ZN
IF (ISURF .EQ. 1) GO TO 75
XT(IX) = XN*UL + UI
GO TO 76
75 XT(IX) = XN* CHX + UI + (UL*UI)*YN
76 CONTINUE
IF (INORM .LE. 0) GO TO 79
GO TO (81, 82, 83, 84), INORM
C INORM = 0 IX = 4, IY = 5, IZ = 6
79 XX = XT(IX)
SINPH = SIN(XT(IZ))
COSPH = COS(XT(IZ))
YY = XT(IY)*SINPH
ZZ = -XT(IY)*COSPH
VT(1) = FI(2)
```

VAL2E

VAL2E

VAL2	141
VAL2	142
VAL2	143
VAL2	144
VAL2	145
VAL2	146
VAL2	147
VAL2	148
VAL2	149
VAL2	150
VAL2	151
VAL2	152
VAL2	153
VAL2	154
VAL2	155
VAL2	156
VAL2	157
VAL2	158
VAL2	159
VAL2	160
VAL2	161
VAL2	162
VAL2	163
VAL2	164
VAL2	165
VAL2	166
VAL2	167
VAL2	168
VAL2	169
VAL2	170
VAL2	171
VAL2	172
VAL2	173
VAL2	174
VAL2	175

```

VT(2) = FI(3)*SINPH + FI(4)*COSPH
VT(3) = FI(4)*SINPH - FI(3)*COSPH
GO TO 90

```

```

C
C INORM = 1 IX = 1, IY = 2, IZ = 3
  A1 XX = XT(IX)
    YY = XT(IY)
    ZZ = XT(IZ)
  110 VT(1) = FI(2)
    VT(2) = FI(3)
    VT(3) = FI(4)
    GO TO 90

```

```

C
C INORM = 2 IX = 1, IY = 3, IZ = 2
  B2 XX = XT(IX)
    YY = XT(IZ)
    ZZ = XT(IY)
  111 VT(1) = FI(2)
    VT(2) = FI(4)
    VT(3) = FI(3)
    GO TO 90

```

```

C
C INORM = 3 IX = 2, IY = 3, IZ = 1
  B3 XX = XT(IZ)
    YY = XT(IX)
    ZZ = XT(IY)
  112 VT(1) = FI(4)
    VT(2) = FI(2)
    VT(3) = FI(3)
    GO TO 90

```

```

C
C INORM = 4 IX = 4, IY = 6, IZ = 5
  B4 XX = XT(IX)
    SINPH = SIN(XT(IY))
    COSPH = COS(XT(IY))

```

VAL2E

VAL2E

VAL2	176
VAL2	177
VAL2	178
VAL2	179
VAL2	180
VAL2	181
VAL2	182
VAL2	183
VAL2	184
VAL2	185
VAL2	186
VAL2	187
VAL2	188
VAL2	189
VAL2	190
VAL2	191
VAL2	192
VAL2	193
VAL2	194
VAL2	195
VAL2	196
VAL2	197
VAL2	198
VAL2	199
VAL2	200
VAL2	201
VAL2	202
VAL2	203

```

YY = XT(IZ)*SINPH
ZZ = XT(IZ)*COSPH
EMY(I) = EMY(I)/ZN
113 VT(1) = FI(2)
    VT(2) = FI(4)*SINPH + FI(3)*COSPH
    VT(3) = FI(4)*COSPH + FI(3)*SINPH
C
90 IF (IC .EQ. 0) GO TO 100
   X(I) = XO + XX*AP11 + YY*AP21 + ZZ*AP31
   Y(I) = YO + XX*AP12 + YY*AP22 + ZZ*AP32
   Z(I) = ZO + XX*AP13 + YY*AP23 + ZZ*AP33
C
C CHECK COORDINATE SYSTEM FOR DIRECTION COSINES
100 IF (IT .EQ. 1) GO TO 500
120 EMX(I) = VT(1)*AP11 + VT(2)*AP21 + VT(3)*AP31
    EMY(I) = VT(1)*AP12 + VT(2)*AP22 + VT(3)*AP32
    EMZ(I) = VT(1)*AP13 + VT(2)*AP23 + VT(3)*AP33
    GO TO 500
C
C POINT NOT WITHIN BOUNDARIES. SET FLAG AND GO TO NEXT POINT.
60 INT(I) = 0
C
C 500 CONTINUE
C
C RETURN
END

```

VAL2E

```

C
C
SURROUTINE SFNTR2
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 COMMON/SURFN2/INORM, ISURF, IX, IY, IZ, U1, UJ, V1, VL, CP, CT, CHX, HP, N3,
      XO, YO, ZO, AP11, AP12, AP13, AP21, AP22, AP23, AP31, AP32, AP33,
2 FLOWC(7), XI(500), YI(500), B(503,7), XKF(503), AA(5999),
      A XB(4), YB(4), ZB(4)
3 NDSET, IABSFT, IP, LORG(20), ISR(20), NS, ISFR, IVIS, IFTYP, IFLOW,
4 BX(2,20), BY(2,20), NB
C
      DIMENSION TITLE(15), E4(12), E48(12), E1(25), LOSF(5),
1 FLOW(503, 7),
      XT(6), VTRA), BP(2)
C
      INTEGER ERROR, PAGE, CASE
      INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
C
      EQUIVALENCE (FLOW(1,1), B(1,1))
C
      DATA RC, NX, MX, MX1, KO
1 /0.1745329E-1, 503, 7, 1, 5999/
C SET CHECKOUT PRINT FLAG
      IPPRINT = 1
C
C INITIALIZE CONSTANT ARRAY FLOWC
      DO 98 I = 1, 7
98 FLOWC(I) = 0.0
C
      IC = 0
      NB = 0
C
C CYCLE ON SUB-REGIONS

```

SFN2 001
 SFN2 002
 SFN2 003
 SFN2 004
 SFN2 005
 SFN2 006
 SFN2 007
 SFN2 008
 SFN2 009
 SFN2 010
 SFN2 011
 SFN2 012
 SFN2 013
 SFN2 014
 SFN2 015
 SFN2 016
 SFN2 017
 SFN2 018
 SFN2 019
 SFN2 020
 SFN2 021
 SFN2 022
 SFN2 023
 SFN2 024
 SFN2 025
 SFN2 026
 SFN2 027
 SFN2 028
 SFN2 029
 SFN2 030
 SFN2 031
 SFN2 032
 SFN2 033
 SFN2 034
 SFN2 035

```

DO 600 II = 1, NS
ISPR = ISR(II)
IG10 = LOGR(IM) + 4 + ISPR
CALL READMS(10, E1, 25, IG10)
READ(10, IG10) E1
LOFF = E1(1) + 0.01
N1 = E1(6) + 0.01
C

DO 500 J = 1, 5
500 LOSF(J) = E1(J+20) + 0.01
IF (ISFR .EQ. 0) GO TO 510
C
IF (LOSF(ISFP) .LE. 0) GO TO 600
C
C READ IN SECONDARY FLOW POINTERS
IG10 = LOSF(ISFR)
CALL READMS(10, E1, 25, IG10)
READ(10, IG10) E1
LOFF = E1(1) + 0.01
N1 = E1(6) + 0.01
C
C READ IN THE DATA
510 ITAG10 = LOFF
IBSF = 0
IB = 0
DO 310 I = 1, N1
CALL READMS(10, E4, 12, ITAG10)
READ(10, ITAG10) E4
ITAG10 = ITAG10 + 1
C
C TRANSFORM DATA
301 CONTINUE
XX = E4(1) * XD
YY = E4(2) * YD
ZZ = E4(3) * ZD

```

SFN2 036
SFN2 037
SFN2 038
SFN2 039C
SFN2 040I
SFN2 041
SFN2 042
SFN2 043
SFN2 044
SFN2 045
SFN2 046
SFN2 047
SFN2 048
SFN2 049
SFN2 050
SFN2 051
SFN2 052C
SFN2 053I
SFN2 054
SFN2 055
SFN2 056
SFN2 057
SFN2 058
SFN2 059
SFN2 060
SFN2 061
SFN2 062C
SFN2 063I
SFN2 064
SFN2 065
SFN2 066
SFN2 067
SFN2 068
SFN2 069
SFN2 070

SFN2 071
 SFN2 072
 SFN2 073
 SFN2 074
 SFN2 075
 SFN2 076
 SFN2 077
 SFN2 078
 SFN2 079
 SFN2 080
 SFN2 081
 SFN2 082
 SFN2 083
 SFN2 084
 SFN2 085
 SFN2 086
 SFN2 087
 SFN2 088
 SFN2 089
 SFN2 090
 SFN2 091
 SFN2 092
 SFN2 093
 SFN2 094
 SFN2 095
 SFN2 096
 SFN2 097
 SFN2 098
 SFN2 099
 SFN2 100
 SFN2 101
 SFN2 102
 SFN2 103
 SFN2 104
 SFN2 105

```

C
  XT(1) = XX*AP1 + YY*AP12 + ZZ*AP13
  XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
  XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
  XT(4) = XT(1)
  XT(5) = SQRT(XT(2)**2 + XT(3)**2)
  IF (XY(2) .EQ. 0.0) GO TO 304
  XT(6) = ATAN2(XT(2),XT(3))
  GO TO 305
304 XT(6) = 0.0
  IF (XT(3) .GT. 0.0) XT(6) = 3.141592654
305 CONTINUE
  IF (IRSF .EQ. 1) GO TO 523

C
C   NORMALIZE THE DATA
  IC = IC + 1
  IF (IC .LE. 500) GO TO 303
  WRITE (TAPEOT,302)
302 FORMAT (1H0,4RH*ROUTINE SFNTRP HAS ATTEMPTED TO LOAD MORE DATA ,
  2 ,/1H ,52H*CALCULATIONS WILL CONTINUE WITH ONLY THE FIRST 100 ,
  3 7HPPOINTS. )
  IC = IC - 1
  GO TO 311
303 YI(IC) = (XT(IY) - VI)/YL
  IF (ISURF .EQ. 1) GO TO 306
  XI(IC) = (XT(IX) - UI)/UL
  GO TO 307
306 CHX = CR + (CT*CR)*YI(IC)
  XLE = UL + (UL*UI)*YI(IC)
  XI(IC) = (XT(IX) - XLE)/CHX

C
307 CONTINUE
C   CHECK IF WITHIN SURFACE BOUNDARIES,
  IF (XI(IC) .LT. 0.0) GO TO 308

```

SFN2 106
 SFN2 107
 SFN2 108
 SFN2 109
 SFN2 110
 SFN2 111
 SFN2 112
 SFN2 113
 SFN2 114
 SFN2 115
 SFN2 116
 SFN2 117
 SFN2 118
 SFN2 119
 SFN2 120
 SFN2 121
 SFN2 122
 SFN2 123
 SFN2 124
 SFN2 125C
 SFN2 126Y
 SFN2 127
 SFN2 128C
 SFN2 129I
 SFN2 130
 SFN2 131
 SFN2 132
 SFN2 133
 SFN2 134
 SFN2 135
 SFN2 136
 SFN2 137
 SFN2 138
 SFN2 139
 SFN2 140

```

IF (XI(IC) .GT. 1.0) GO TO 308
IF (VI(IC) .LT. 0.0) GO TO 308
IF (YI(IC) .GT. 1.0) GO TO 30A

C POINT WITHIN SURFACE BOUNDARIES. SET UP FLOW BOUNDARIES --
C -- INITIALIZE AS SURFACE BOUNDARIES.
IF (IB .EQ. 1) GO TO 535
IB = 1
NB = NB + 1
AX(1,NB) = 0.0
BY(1,NB) = YI(IC)
IF (JSFR .EQ. 0) GO TO 520
BX(1,NB) = XI(IC)

C 520 CONTINUE
IF (IFTYP .EQ. 2) GO TO 530
IF (ISFR .EQ. 5) GO TO 530
IGIOSF = LUSF(ISFR+1)
IF (IGIOSF .LE. 0) GO TO 530
CALL READMS(10,E1,25,IGIOSF)
READ(10*IGIOSF) E1
LOFF = E1(1) + 0.01
CALL PEADMS(10,E4B,12,LOFF)
READ(10*LOFF) E4B
E4(1) = E4B(1)
E4(2) = E4B(2)
E4(3) = E4B(3)
IBSF = 1
GO TO 301

523 IBSF = 0
YIR = (XT(IY) - VI)/VL
IF (JSURF .EQ. 1) GO TO 526
XIR = (XT(IX) - UI)/UL
GO TO 527

526 CHR = CR + (CT*CR)*YIB
    
```

SFN2RP

SFN2 141
SFN2 142
SFN2 143
SFN2 144
SFN2 145
SFN2 146
SFN2 147
SFN2 148
SFN2 149
SFN2 150
SFN2 151
SFN2 152
SFN2 153
SFN2 154
SFN2 155
SFN2 156
SFN2 157
SFN2 158
SFN2 159
SFN2 160
SFN2 161
SFN2 162
SFN2 163
SFN2 164
SFN2 165
SFN2 166
SFN2 167
SFN2 168
SFN2 169
SFN2 170
SFN2 171
SFN2 172
SFN2 173
SFN2 174
SFN2 175

XLR = U1 + (UL*U1)*YIB
XIR = (XT(IX) + XLB)/CHB

C

527 CONTINUE

AX(2,NB) = XIR
BY(2,NB) = YIB
GO TO 535

C

C

C

530 CONTINUE

AX(2,NB) = 1.0
BY(2,NB) = BY(1,NB)
C POINT WITHIN BOUNDARIES. SET UP FLOW ARRAYS.
535 CONTINUE

C

C

C TRANSFORM FLOW DIRECTION COSINES

VT(1) = E4(8)*AP11 + E4(9)*AP12 + E4(10)*AP13
VT(2) = E4(8)*AP21 + E4(9)*AP22 + E4(10)*AP23
VT(3) = E4(8)*AP31 + E4(9)*AP32 + E4(10)*AP33
SINPH = SIN(XT(6))
COSPH = COS(XT(6))
VT(4) = VT(1)
VT(5) = VT(2)*SINPH + VT(3)*COSPH
VT(6) = VT(2)*COSPH + VT(3)*SINPH
FLOW(IC,1) = E4(7)
FLOW(IC,2) = VT(IX)
FLOW(IC,3) = VT(IY)
FLOW(IC,4) = VT(IZ)
FLOW(IC,5) = E4(11)
FLOW(IC,6) = E4(12)
FLOW(IC,7) = XT(IZ)
GO TO 310

C

C POINT NOT WITHIN BOUNDARIES. RESET COUNTER AND GO TO NEXT POINT.

SFN2RP

SFN2 176
 SFN2 177
 SFN2 178
 SFN2 179
 SFN2 180
 SFN2 181
 SFN2 182
 SFN2 183
 SFN2 184
 SFN2 185
 SFN2 186
 SFN2 187
 SFN2 188
 SFN2 189
 SFN2 190
 SFN2 191
 SFN2 192
 SFN2 193
 SFN2 194
 SFN2 195
 SFN2 196
 SFN2 197
 SFN2 198
 SFN2 199
 SFN2 200
 SFN2 201
 SFN2 202
 SFN2 203
 SFN2 204
 SFN2 205
 SFN2 206
 SFN2 207
 SFN2 208
 SFN2 209
 SFN2 210

```

C 308 IC = IC + 1
C 310 CONTINUE
C 600 CONTINUE
    IF (IC .GT. 2) GO TO 311
    WRITE(TAPEOT,1000) IC
C 1000 FORMAT(1M1, 5XONLY ,I3,29H POINTS WERE FOUND IN SFNTR2 ,
    1 56H TO BE WITHIN SURFACE BOUNDARIES. THIS IS INSUFFICIENT /
    2 55H TO USE THE SURFACE SPLINE. RUN CONTINUES, BUT USER IS,
    3 36H CAUTIONED TO CHECK RESULTS CAREFULLY.)
    WRITE(TAPEOT,1005) NDSET,IABSFT,IR,YSR,ISFR
C 1005 FORMAT(1M0,40H THIS APPLIES TO THE FOLLOWING FLOW DATA /
    1 12H DATA SET = ,I3, 10X17HALPHA-BETA SET = ,J3,
    2 10X14H FLOW REGION = ,I3/
    3 15H SUB-REGIONS = , 20I3/
    4 14H SECONDARY FLOW = ,I3)
    ERRPR = 1
    RETURN
C C ALL THE SURFACE DATA ARE IN PROPER ARRAYS.
C C 311 N2 = IC
C C CHECK BOUNDARY POINTS
    IF (NB .GT. 1) GO TO 700
    NB = 2
    BX(1,2) = BX(1,1)
    RX(2,2) = BX(2,1)
    BY(1,2) = 1.0
    RY(2,2) = 1.0
C C NORMALIZE TO FLOW BOUNDARIES (IN XI ONLY):
C C FIRST ARRANGE BOUNDARY DATA IN ORDER OF ASCENDING YI VALUES
C 700 IB = 0
C 710 IB = IB + 1
    
```

8FN2 211
 8FN2 212
 8FN2 213
 8FN2 214
 8FN2 215
 8FN2 216
 8FN2 217
 8FN2 218
 8FN2 219
 8FN2 220
 8FN2 221
 8FN2 222
 8FN2 223
 8FN2 224
 8FN2 225
 8FN2 226
 8FN2 227
 8FN2 228
 8FN2 229
 8FN2 230
 8FN2 231
 8FN2 232
 8FN2 233
 8FN2 234
 8FN2 235
 8FN2 236
 8FN2 237
 8FN2 238
 8FN2 239
 8FN2 240
 8FN2 241
 8FN2 242
 8FN2 243
 8FN2 244
 8FN2 245

```

    I1 = 1
    720 YY = BY(IB,I1)
    I2 = I1 + 1
    J = 0
    DO 730 I = I2,NB
    IF (YY .LT. BY(IR,I)) GO TO 730
    J = J + 1
    YY = BY(IR,I)
    730 CONTINUE
C
    IF (J .EQ. 0) GO TO 740
    BY(IB,J) = BY(IR,I)
    BY(IR,I) = YY
C
    740 I1 = I1 + 1
    IF (I1 .LT. NB) GO TO 720
    IF (IB .EQ. 1) GO TO 710
C
C BOUNDARY DATA NOW IN ORDER. PROCEED WITH NORMALIZATION.
    IF (TPRINT .EQ. 0) GO TO 770
    WRITE(TAPEOT,750) NB
    750 FOPMAT(1H1,18HBOUNDARY DATA, , I3, AM.POINTS./
    1 1H0, 3X1H1, 7X2HX1, 12X2HY1, 12X2MX2, 12X2HY2//)
    WRITE(TAPEOT,760) (I,BX(1,I),BY(1,I),RX(2,I), I=1,NB)
    760 FOPMAT(1H , I3, 4F14.6)
    770 CONTINUE
    IC = 0
    DO 800 I = 1,N2
    XX = XI(I)
    YY = YI(I)
    IB = 0
    800 IB = IB + 1
C
    DO 810 I1 = 1,NB
    J = I1
  
```

SFN2 246
 SFN2 247
 SFN2 248
 SFN2 249
 SFN2 250
 SFN2 251
 SFN2 252
 SFN2 253
 SFN2 254
 SFN2 255
 SFN2 256
 SFN2 257
 SFN2 258
 SFN2 259
 SFN2 260
 SFN2 261
 SFN2 262
 SFN2 263
 SFN2 264
 SFN2 265
 SFN2 266
 SFN2 267
 SFN2 268
 SFN2 269
 SFN2 270
 SFN2 271
 SFN2 272
 SFN2 273
 SFN2 274
 SFN2 275
 SFN2 276
 SFN2 277
 SFN2 278
 SFN2 279
 SFN2 280

```

        IF (YY .LT. BY(IB,II)) GO TO 820
    C 610 CONTINUE
    C 820 IF (J .EQ. 1) J = 2
        DY = BY(IB,J) - BY(IB,J-1)
        IF (DY .EQ. 0.0) GO TO 900
        DX = BX(IB,J) - BX(IB,J-1)
        BP(IB) = BX(IB,J-1) + (YY - BY(IB,J-1))*DX/DY
        GO TO (830,840), IA
    C 830 IF (XX .LT. BP(IA)) GO TO 900
        GO TO 800
    C 840 IF (XX .GT. BP(IA)) GO TO 900
    C 850 POINT WITHIN FLOW BOUNDARIES
        IC = IC + 1
        XI(IC) = XX - BP(1)
        YI(IC) = YY
        IF (IYIS .EQ. 0) GO TO 850
        XI(IC) = XI(IC)/(BX(2,1) - BX(1,1))
        GO TO 860
    C 850 XI(IC) = XI(IC)/(BP(2) - BP(1))
    C 860 FLOW(IC,1) = FLOW(I,1)
        FLOW(IC,2) = FLOW(I,2)
        FLOW(IC,3) = FLOW(I,3)
        FLOW(IC,4) = FLOW(I,4)
        FLOW(IC,5) = FLOW(I,5)
        FLOW(IC,6) = FLOW(I,6)
        FLOW(IC,7) = FLOW(I,7)
    C 900 FLOWC(II) = FLOWC(II) + FLOW(IC,II)
    C 900 CONTINUE
        IF (IC .GT. 2) GO TO 910
        WRITE(TAPEOT,1010) IC
    
```

```

1010 FORMAT(IH1, 5HCNLY ,I3, 29M POINTS WERF FOUND IN SFNTR2,
15IHTO BE WITHIN FLOW BOUNDARIES. THIS IS INSUFFICIENT/
2 55M TO USE THE SURFACE SPLINE. RUN CONTINUFS. BUT USER IS,
3 34M CAUTIONED TO CHECK RESULTS CAREFULLY.)
WRITE(TAPEOT,1005) NDSET,IABSET,IR,ISR,ISFR
ERROR = 1
RETURN
910 CONTINUE
N2 = IC
N3 = N2 + 3
C PRINT NORMALIZED ARRAYS
IF (IPRINT .EQ. 0) GO TO 330
WRITE(TAPEOT,3060) NDSET, IABSFT,IR
3060 FORMAT(IH1,34MNORMALIZED SURFACE DATA. NDSET = ,I2, 5X,
1 9HIABSET = ,I2, 5X, 9HREGION = ,I2/I40. 74,IHI, T13,2HXI,
2 T27,2HYI, T39,4HMACH, T55,2MXX, T69,2HMY, T83,2HMZ,
3 T95,4HP/P1, T105,4HT/T1, T125,2HZI//)
WRITE(TAPEOT,3070) (I,XI(I), YI(I), (FLOW(I,J),J=1,7), I=1,N2)
3070 FORMAT(1M , I3, 9F14,6)
C
C
C 330 CONTINUE
C
C 350 J = 1,7
FLOWC(J) = FLOWC(J)/N2
C 340 I = 1,N2
340 FLOW(I,J) = FLOW(I,J) - FLOWC(J)
350 CONTINUE
C
C
C CALCULATE COEFFICIENT ARRAY (SURFACE SPLINE).
LS = 0
CALL POWFH1 (N2,MX,XI,YI,FLOW,KKF,NX,MX,LS,TAPEB)
C
CALL SOLVIT(AA,N3,MX,KD,TAPER,TAPEC,TAPED,TAPEB,B,NX,MX,NERR)

```

SFN2RP

SFN2 316
SFN2 317
SFN2 318
SFN2 319
SFN2 320
SFN2 321
SFN2 322
SFN2 323
SFN2 324
SFN2 325
SFN2 326
SFN2 327
SFN2 328
SFN2 329
SFN2 330
SFN2 331
SFN2 332
SFN2 333

```

C      IF (NERR, NE, 0) WRITE(TAPEOT,220)
      220 FORMAT(1H, '16MATRIX SINGULAR,')
C
C      COEFFICIENTS ARE IN ARRAY R(I,J)
      IF (IPRINT, EQ, 0) GO TO 400
      WRITE(TAPEOT,3080)
      3080 FORMAT(1H, '42HSURFACE INTERPOLATION COEFFICIENTS R(I,J)')
      WRITE(TAPEOT,3090) (I, (R(I,J), J=1,7), I=1,N3)
      3090 FORMAT(1H, '15, '15.5)
      WRITE(TAPEOT,3100) FLOWC
      3100 FORMAT(1H, '5X, '7F15.5)
      400 CONTINUE
C
C
C      RETURN
      END
```

SFN2RP

ERF

```

C
C  THIS FUNCTION ROUTINE IS REQUIRED BY CDC MODELS OF THE ARBITRARY-
C  BODY PROGRAM.  IT IS USED WITH THE ERROR FUNCTION SUBPROGRAM (ERRF).
C  THESE ROUTINES ARE NOT REQUIRED IN THE IBM/360 MODEL BECAUSE THEY
C  ARE ALREADY AVAILABLE IN THE 360 FORTRAN LIBRARY.
      CALL ERRF(X,Y)
      ERF = Y
      RETURN
      END

```

```

ERF 001C
ERF 002
ERF 003
ERF 004
ERF 005
ERF 006
ERF 007C
ERF 008C
ERF 009C
ERF 010C

```

ERF

ERRF

ERRF 001C
ERRF 002
ERRF 003
ERRF 004
ERRF 005
ERRF 006C
ERRF 007C
ERRF 008C
ERRF 009C
ERRF 010C
ERRF 011C
ERRF 012C
ERRF 013C
ERRF 014C
ERRF 015C
ERRF 016C
ERRF 017C
ERRF 018C
ERRF 019C
ERRF 020C
ERRF 021C

```
C SURROUTINE ERRF(X,Y)
C THIS ROUTINE IS REQUIRED BY CDC MODELS OF THE ARBITRARY BODY
C PROGRAM. THIS ROUTINE IS NOT REQUIRED IN THE IBM/360 MODEL
C BECAUSE IT IS ALREADY AVAILABLE IN THE IBM FORTRAN LIBRARY,
C
C DIMENSION C(Y)
C DATA (C(J),J=1,7)/1.,7.05230794E-2,4.22820123E-2,9.2705272E-3,
1 1.520143E-4,2.765672E-4,4.30638E-5/
C IF (X.EQ. 0.0) GO TO 70
C Z = ABS(X)
C Y = SIGN(1.0,X)
C IF (Z.GE. 4.17) RETURN
C P = C(7)
C DO 50 J=1,6
C X = 7 - J
50 P = C(K) * P * Z
C Y = Y * (1.0 - P ** (.16))
C RETURN
70 Y = 0.0
C END
```

ERRF

ACONE

```

C SUBROUTINE ACONE(ANGLE, CP, ALF, PHI, ALFP, EMP, IT, IPPCK)
C CIRCULAR CONE AT ANGLE OF ATTACK.
C BASED ON ARC CP #792 AND EXTENDED BY D.N. SMYTH.

```

```

C DIMENSION FS(8), BS(8), ANGLE(3)
C DIMENSION AA(6), A(18)

```

```

C COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYP(2)
C COMMON/FSBS/ FS,BS

```

```

C DATA A/ 0.07657, 1.4775, 0.063669,
2 0.42339, 0.13241, 0.035871,
3 0.002083, 0.075797, 0.01923,
4 0.29898, 0.10011, 0.29589,
5 -0.99727, -0.41751, 0.068791,
6 0.030442, 0.10422, 0.063801/
C DATA RC/.17453292E=1/

```

```

C G = GAM
C EM = FS(6)
C GM1 = G * 1.0
C DCP = ANGLE(1)*RC
C GP1 = G + 1.0
C PMJN = 0.3
C CPCONE = 0.0
C CPSTAG = 0.0
C CPX = 0.0
C DESK = 0.0
C CGSA = COS(ALF)
C SINA = SIN(ALF)
C COSP = COS(PHI)
C SINP = SIN(PHI)
C EM2 = EM**2
C CPMIN = (PMIN * 1.0)/(0.5*G*EM2)

```

```

ACON 001
ACON 002
ACON 003
ACON 004
ACON 005
ACON 006
ACON 007
ACON 008
ACON 009
ACON 010
ACON 011
ACON 012
ACON 013
ACON 014
ACON 015
ACON 016
ACON 017
ACON 018
ACON 019
ACON 020
ACON 021
ACON 022
ACON 023
ACON 024
ACON 025
ACON 026
ACON 027
ACON 028
ACON 029
ACON 030
ACON 031
ACON 032
ACON 033
ACON 034
ACON 035

```

ACONE

ACONE

```

C
FS(6) = FS6
5 IF (ALF .LT. 1.E=4) GO TO 40
C
C NORMAL FLOW COMPONENT
EMC2 = EM2*(SINA)**2
IF (EMC2 = 1.0) 10, 10, 20
C SUBSONIC NORMAL COMPONENT
10 PSTAG = (1.0 + 0.5*GM1*EMC2)**(G/GM1)
GO TO 30
C SUPERSONIC NORMAL COMPONENT
20 PSTAG = (0.5*GP1*EMC2)**(G/GM1)*(GP1/(2.*G*EMC2-GM1))**(1./GM1)
30 CPSTAG = (FSTAG = 1.0)/(0.5*G*EM2)*COSP*ABS(COSP)
C
IF (CPSTAG .LT. 0.0 ) CPSTAG = 0.0
IF (DCR .LT. 1.E=4) GO TO 40
C CROSS PRODUCT TERM = EMPIRICAL FIT FROM CD #792.
SDN = SIN(DCR)*COS(DCR)
X = 0.5*3.1415927/(SCD*SQRT(EM2=1.0))
BK = 1.95 + 0.07*COS(X)
CPX = 2.0*BK*SCD*COASP*SINA*COXA
GO TO 40
C
C CONF FLOW IS DETACHED OR UNDEFINED (DCR .LT. 0.0)
C USE IMPACT METHODS
C
C FIRST GET IMPACT ANGLE (IN MERIDIAN PLANE)
35 CONTINUE
DELI = ALFP + DCR
IF (DELI .GT. 0.0) GO TO 36
C
C EXPANSION FLOW
ANGLE(2) = ARS(DELI)*57.29577951
FS(6) = EMP
IPRINT = 0

```

```

ACON 071
ACON 072
ACON 073
ACON 074
ACON 075
ACON 076
ACON 077
ACON 078
ACON 079
ACON 080
ACON 081
ACON 082
ACON 083
ACON 084
ACON 085
ACON 086
ACON 087
ACON 088
ACON 089
ACON 090
ACON 091
ACON 092
ACON 093
ACON 094
ACON 095
ACON 096
ACON 097
ACON 098
ACON 099
ACON 100
ACON 101
ACON 102
ACON 103
ACON 104
ACON 105

```

ACONE

ACONE

```

ISDET = 2
CALL EXPAND(ANGLE, MER, IPRINT, ISDET, CP)
EMSURF = BS(A)
EM2 = EMP**2
PSURF = 0.5*G*EM2*CP + 1.0
IF (PSURF .LT. PMIN) PSURF = PMIN
GO TO 50

C
C COMPRESSION FLOW, USE TANGENT-CONE EMPIRICAL
36 CK = 2.*GPI/(G+3.)*EMP*SIN(DEL I)
EMNS = (CK + EXP(-CK))**2
CP = (.8.*GPI*EMNS/((3.+5.)*EMNS + 2.))*STN(DEL I)**2
PSURF = 0.5*G*EM2*CP + 1.0
IF (PSURF .LT. PMIN) PSURF = PMIN
PCONE = PSURF
DPSK = PCONE * (2.*G*EMNS = GPI)/GPI
GO TO 50

C
C COMBINED SURFACE PRESSURE
40 PSURF = 0.5*G*EM2*(CPCONE + CP9TAG + CPX) + 1.0

C
C LIMIT MINIMUM PSURF
IF (PSURF .LT. PMIN) PSURF = PMIN

C
C SHOCK ANGLE DSR, RELATIVE TO AXIS
C
C USE IMPACT METHODS TO OBTAIN DPSK
C (ADJUST TO ALPHA = ZERO RESULTS)
C
DEL I = DCR
IS = 1
FDP = 1.0
FMC = EMA
GO TO 46

```

ACON 106
ACON 107
ACON 108
ACON 109
ACON 110
ACON 111
ACON 112
ACON 113
ACON 114
ACON 115
ACON 116
ACON 117
ACON 118
ACON 119
ACON 120
ACON 121
ACON 122
ACON 123
ACON 124
ACON 125
ACON 126
ACON 127
ACON 128
ACON 129
ACON 130
ACON 131
ACON 132
ACON 133
ACON 134
ACON 135
ACON 136
ACON 137
ACON 138
ACON 139
ACON 140

ACONE

ACONE

```

C   CALCULATE VALUE FROM NCONF
45 CONTINUE
   EMNS = (EMASIN(DSC))**
   PS = (2.*G*EMNS + GM1
   FDP = PS/P2
   EMC = EMP

C   NOW CALCULATE VALUE AT ... IMPACT ANGLE
   DELI = ALFP + DCR
   DPSK = 0.0
   IF (DELI .LE. 0.0) GO TO 50
   I3 = 2
46   EMD = EMC*SIN(DELI)
   CK = 2.*GPI/(G+3.)*EMD
   EMNS = (CK + EXP(-CK))**2
   P2 = (2.*G*EMNS + GM1)/GPI
   GO TO (45,47), I3
47   DPSK = PSURF - P2*FDP
50   IF (DPSK .LT. 0.0) DPSK = 0.0
   PSHK = PSURF = DPSK
   IF (PSHK .LT. 1.0) PSMK = 1.0
C   SURFACE MACH NUMBER
   EMSURF = SQRT((EM2*(GPI*PSHK+GM1) + 2.*GPI*GM1)
1     / (PSHK*(GM1*PSHK + GPI)))
C
   EM2 = EMP**2
   DSR = (GPI*PSHK + GM1)/(2.*G*EM2)
   IF (DSR .GT. 1.0) DSR = 1.0
   YF (DSR .LT. 0.0) DSR = 1.0/EM2
   DSR = ARSIN(SQRT(DSR))
   DSR = DSR + ALFP

C   CALCULATE BS ARRAY
   FM2 = EM**2

```

ACON 141
ACON 142
ACON 143
ACON 144
ACON 145
ACON 146
ACON 147
ACON 148
ACON 149
ACON 150
ACON 151
ACON 152
ACON 153
ACON 154
ACON 155
ACON 156
ACON 157
ACON 158
ACON 159
ACON 160
ACON 161
ACON 162
ACON 163
ACON 164
ACON 165
ACON 166
ACON 167
ACON 168
ACON 169
ACON 170
ACON 171
ACON 172
ACON 173
ACON 174
ACON 175

ACONE

ACONE

```
BS(6) = EMSURF
BS(2) = FS(2)*PSURF
CP = (PSURF = 1.0)/(0.5*G*EM2)
ANGLE(3) = DSR/RC
TST1 = (1. + 0.5*GM1*EM2)/(1. + 0.5*GM1*EMSURF**2)
RS(3) = FS(3)*TST1
RS(1) = FS(1)*PSURF/TST1
RS(4) = FS(4)*SORT(TST1)
IF (RS(3).GE.AV1) BS(5) = 2.3*E-8*BS(3)**1.5/(BS(3)+198.6)
IF (RS(3).LT.AV1) BS(5) = AV1*BS(3)**AV3
RS(7) = BS(6)*BS(4)
RS(8) = BS(1)*RS 7)/BS(5)

IF (IPRCK .NE. 1) RETURN
WRITE(6,100) CPCONE,CPSTAG,CPX,CP
100 FORMAT(1H0, 9HCPCONE = ,F10.5, 5X9HCPSTAG = ,F10.5,
1 5X6HCPX = ,F10.5, 5X5HCP = ,F10.5/)
WRITE(6,200) FS,BS
200 FORMAT(1H ,10HFS ARRAY , 8F15.6/1H ,10HBS ARRAY , 8F15.6)
RETURN
END
```

C

ACONE

VISCUS

```

OVERLAY (MARK4,2,4)
PROGRAM VISCUS
SUBROUTINE VISCUS

COMMON /EXEC/CASE, TITL, PAGE, ERROR
COMMON /GDATA/LTOT, J, SYMFCT, IORN, IGTYPF, L
COMMON /FLIGHT/MACH, ALT, PSTAG, V, RFND, PFS, TFS, AFS, RHOFS, VIS
COMMON /TAPE/TAPFIN, TAPECT, TAPFA, TAPEB, TAPFC, TAPFD, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /ABDATA/N4B, ALPHA(20), BETA(20), POI(20), CDLTA(20), DI(20),
      PI(20), PJ(20)
COMMON /REF/SRFF, MAC, SPAN, XCG, YCG, ZCG
COMMON /FARRAY/F
COMMON /GASD/GAM, GASC, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
COMMON /SKIND/LS, IS(100,9), SURF(100,R), DELTA, SWEAP
COMMON /C1 /INORM, ISURF, IX, IY, IZ, UI, IUL, VI, VL, CR, CT, CHX, NZ, NS,
      X0, Y0, Z0, AP11, AP12, AP13, AP21, AP22, AP23, AP31, AP32, AP33,
      FLOWC(2), XY(500), YI(500), B(S03,2), XKF(503), AA(2477),
      X8(4), Y8(4), Z8(4),
      NDSFT, IARSFT, IR, LURG(20), ISR(20), N4, TSFR, IVIS, IFTYP, IFL04,
      RX(2,20), RY(2,20), N6, DUM(4)
DIMENSION F(11), IPANL(20), IPR(20), ISF(20), ISA(20)
DIMENSION EP(25), EI(8), E4(41), E2(11),
      FLM(25), TITLE(15), COM(10), F(25), E5(41), F6(41), E7(11), F8(14)
DIMENSION LCOM(41), LSUM(41), TITLE9(5)
DIMENSION IGAS(1000)
DIMENSION LTOTAR(20)
INTEGER ERROR, SYMFCT, PAGE, CASE
INTFGER TAPEFIN, TAPECT, TAPEA, TAPEB, TAPFC, TAPEB, TAPED, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 REAL MACH, MAC
DATA ISIZE, ISIZ2/1000, 100/
DATA TITLE9/10HFORCE DATA, 10H SAVE UNIT, 10H
1 10H , 10H

```

```

VISC 001C
VISC 002C
VISC 003I
VISC 004
VISC 005
VISC 006
VISC 007
VISC 00P
VISC 009
VISC 010
VISC 011
VISC 01P
VISC 013
VISC 014
VISC 014
VISC 016
VISC 017
VISC 01A
VISC 019
VISC 020
VISC 021
VISC 022
VISC 023
VISC 028
VISC 025
VISC 026
VISC 027
VISC 028
VISC 029
VISC 030
VISC 031
VISC 032
VISC 033
VISC 034C
VISC 035C

```

VISCUS

C C C

VISCUS

```
C DATA TITLE9/4HFORC,4HE DA,4HTA S,4HUNIT,
C
C SET INITIAL CONSTANTS FOR START OF CASE
C ERROR = 0
C NCON = 0
C I9 = 0
C ISFCWK = 0
C NEWSF = 0
C
C CALL HEADER
C WRITE (TAPE9,10)
C 10 FORMAT (1H0,30H** SKIN FRICTION FORCE PROGRAM)
C
C READ TITLE CARD AND FLAGS
C READ (TAPEIN,20) NCONP,IFSAVE,TITLE
C 20 FORMAT (I2,I1,3X,15A4)
C SET UP CONSTANTS FOR FORCE DATA SAVE ON UNIT 9
C IF (IFSAVE .EQ. 2) GO TO 60
C IF (IFSAVE .NE. 0) GO TO 40
C NTOT = 0
C ISUM = 0
C NEXT = 10
C DO 21 J=1,41
C LCON(I) = 0
C 21 LSUM(I) = 0
C GO TO 50
C
C RETRIEVE PREVIOUSLY SAVED TABLE OF CONTENTS
C 40 TG9 = 1
C CALL READMS (9,E1,R,1,1)
C READ (9,TG9) E1
C NTOT = F1(1)
C ISUM = E1(2)
C NEXT = F1(5)
```

VISC 0361
VISC 037
VISC 038
VISC 039
VISC 040
VISC 041
VISC 042
VISC 043
VISC 044
VISC 045
VISC 046
VISC 047
VISC 048
VISC 049
VISC 050
VISC 051
VISC 052
VISC 053
VISC 054
VISC 055
VISC 056
VISC 057
VISC 058
VISC 059
VISC 060
VISC 061
VISC 062
VISC 063
VISC 064
VISC 065
VISC 066C
VISC 067I
VISC 068
VISC 069
VISC 070

VISCUS

VISCUS

```

P  CHECK IF VALID RECORD
   IF (F1(4) .EQ. TITLE9(1)) GO TO 43
   WRITE (TAPEIN,42)
42  FORMAT (1H0,47H***IFSAVE WAS INPUT = 1 IN PRES VET UNIT 9 HAS ,
      1 55HNEVER BEEN INITIALIZED WITH A IFSAVE = 0. PROGRAM STOP. )
      STOP
43  ICG9 = 2
      CALL READMS (9,5,41,ICG9)
      READ (9,IG9) F5
      DO 45 I=1,40
45   LCOM(I) = E5(I+1)
C
50  DO 22 I=1,20
      E4(I+1) = ALPHA(I)
22   F4(I+21) = HFTA (I)
      F4(I) = NAB
      DO 24 I=1,5
24   E1(I+3) = TITLE9(I)
      DO 25 I=1,41
25   E6(I) = 0.0
      DO 26 I=1,11
26   F7(I) = 0.0
C
C  READ GEOMETRY DATA SOURCE DATA
60  READ (TAPEIN,70) IPANL,ISK,NS
70   FORMAT (20I2,11,13)
      NCOM = NCOM + 1
C  CHECK IF NEW SKIN FRICTION FLAG CARDS ARE TO BE READ
   IF (ISK .GT. 1) GO TO 110
C
C *****
C  READ SKIN FRICTION METHOD DATA (1 CARD OR NAB CARDS)
      DO 100 I=1,NAB
100  IF (ISK.EQ.0 .OR. I.EQ.1) READ (TAPEIN,80) ISEMTM,IPRINT,ISAVE
      80  FORMAT (3I1)

```

VISC 071
VISC 072
VISC 073
VISC 074
VISC 075
VISC 076
VISC 077
VISC 078C
VISC 0791
VISC 080
VISC 081
VISC 082
VISC 083
VISC 084
VISC 085
VISC 086
VISC 087
VISC 088
VISC 089
VISC 090
VISC 091
VISC 092
VISC 093
VISC 094
VISC 095
VISC 096
VISC 097
VISC 098
VISC 099
VISC 100
VISC 101
VISC 102
VISC 103
VISC 104
VISC 105

VISCUS

VISCUS

VISC 106
VISC 107
VISC 108
VISC 109
VISC 110
VISC 111
VISC 112
VISC 113
VISC 114
VISC 115
VISC 116
VISC 117
VISC 118
VISC 119
VISC 120C
VISC 1211
VISC 122
VISC 123
VISC 124
VISC 125
VISC 126
VISC 127
VISC 128
VISC 129
VISC 130
VISC 131
VISC 132
VISC 133C
VISC 1341
VISC 135
VISC 136
VISC 137
VISC 138
VISC 139
VISC 140

```
IPR(I) = IPRINT
ISF(I) = ISFMTH
ISA(I) = ISAVE
IF (ISFMTH .EQ. 1) ISFCHK = 1
IF (ISFMTH .EQ. 0) NEWSF = 1
100 CONTINUE
IF (NEWSF .EQ. 1) REWIND TAPE

C 110 CONTINUE
C *****
C OBTAIN GEOMETRY DATA FROM SAVE UNIT FOR COMPLETE COMPONENT AND
C PLACE IN GEOMETRY DATA WORKING ARRAYS
IG4 = 2
CALL HEADMS (4,E,25,IG4)
READ (4,IG4) E
NEXTG = E(1)
NP = E(2)
NPMAX = E(3)
NRFM = E(4)
K = 0
NCS = 0

C
DO 130 I=1,20
  IF (IPANL(I) .EQ. 0) GO TO 140
  NCS = NCS + 1
  IG4 = IPANL(I)*5
  CALL HEADMS (4,EP,25,IG4)
  READ (4,IG4) EP
  ISTAT3 = EP(1)
  COM(I) = EP(2)
  ISTART = EP(3)
  N = EP(4)
  IORN = EP(5)
  SYMFCY = EP(6)
C
```

VISCUS

VISCUS

```
C
      ISTART = ISTART + NMEM
      DO 120 JI=1,N
        K = K + 1
        IF (K .GT. ISIZE) GO TO 190
        IG4 = ISTART + NMEM*I
        IG4SK(K) = IG4
        IF (NMEMSF .NE. 1) GO TO 120
        CALL READMS(4,ELEM,25,IG4)
        READ(4,IG4) ELEM
        WRITE(TAPEE) IG4,ELEM
      C
      120 CONTINUE
      130 CONTINUE
      140 LTOT = K
      C GEOMETRY DATA LOAD IS COMPLETE
      C
      C
      C CHECK IF SIMPLE ELEMENTAL SKIN FRICTION IS TO BE CALCULATED
      IF (ISFCHK .NE. 1) GO TO 230
      C READ SKIN FRICTION DATA CARDS
      C CHECK IF NUMBER OF ELEMENTS SPECIFIED IS CORRECT (NS = LTOT)
      IF (NS .EQ. LTOT) GO TO 160
      WRITE (TAPEUT,150) NS,LTOT
      150 FORMAT (1H0,49H***MAPPING**THE NUMBER OF SKIN FRICTION SURFACES ,
      2 15)
      STOP
      160 IF (NS .GT. LTOT) GO TO 190
      DO 170 I=1,NS
      C 170 READ (TAPEIN,180,END=210, (IS(I,J),J=1,9), (SURF(I,J),J=1,8)
      170 READ (TAPEIN,180) (IS(I,J),J=1,9), (SURF(I,J),J=1,8)
      IF (EOF(TAPEIN)) 210,230
      180 FORMAT (I2,A11,2F9.0,3F6.0,2F6.0,F4.0)
      C GO TO 230
      190 WRITE (TAPEUT,200) ISIZE,ISIZE
```

VISCUS

VISCUS

```
200 FORMAT (1H :47H*** ERROR *** NUMBER OF ELEMENTS PER COMPONENT ,  
1 14HCANNOT EXCEED ,14,71H ,14X,22MDIVIDE THE SHAPE INTO ,  
2 47HMORE COMPONENTS WITH NO COMPONENT GREATER THAN ,14,  
3 9H ELEMENTS )  
STOP  
210 WRITE (TAPE07,220)  
220 FORMAT (1F0,46H*** ATTEMPT TO READ END OF FILE WHILE READING ,  
1 19HSKIN FRICTION CARDS)  
STOP  
C 230 CONTINUE  
C  
C ISAS = 0  
IF (IFSAVE .EQ. 2) GO TO 235  
NTOT = NTOT + 1  
LCOM(NTOT) = NEXT  
NEXT = NEXT + 2  
235 CONTINUE  
C  
C CHECK FOR NEW INTEGRAL SKIN FRICTION METHOD  
IF (NEWSF .EQ. 0) GO TO 240  
C  
C CALL INTEGRAL BOUNDARY LAYER PROGRAM  
CALL INTEG  
C  
C CALL PRE-PROCESSOR FOR VISCOUS SURFACE DATA  
CALL CFINPT(1SF,LTOTAB)  
C  
C 240 CONTINUE  
C  
C START ALPHA=BETA LOOP (DO ALL ALPHA=BETAS FOR GIVEN COMPONENT)  
DO 250 J=1,NAB  
ISFTH = ISF(J)  
IPRINT = IPR(J)
```

VISC 176
VISC 177
VISC 178
VISC 179
VISC 180
VISC 181
VISC 182
VISC 183
VISC 184
VISC 185
VISC 186
VISC 187
VISC 188
VISC 189
VISC 190
VISC 191
VISC 192
VISC 193
VISC 194
VISC 195
VISC 196
VISC 197
VISC 198
VISC 199
VISC 200
VISC 201
VISC 202
VISC 203
VISC 204
VISC 205
VISC 206
VISC 207
VISC 208
VISC 209
VISC 210

VISCUS

VISCUS

```
ISAVE = I3*(J)
IF (ISAVE .EQ. 1) ISAS = 1

C ***GO TO SKIN FRICTION FORCE PROGRAM
CALL FORSF (NEXTG,ISUS,IPRINT,ISAVE,ISEATH,L10TAB)
IF (ERROR .NE. 0) GO TO 530

C C SAVE FORCE DATA ON UNIT 9
IF (IFSAVE .EQ. 2) GO TO 250
CALL WRITMS (9,F,11,NEXT)
WRITE (9,NEXT) F
NEXT = NEXT + 1
CALL WRITMS (9,E7,11,NEXT)
WRITE (9,NEXT) E7
NEXT = NEXT + 1

C 250 CONTINUE
C ***MENU OF ALPHA BETA LOOP

IF (IFSAVE .EQ. 2) GO TO 450

C C WRITE HEADER INFORMATION FOR FORCE OUTPUT PAGE
CALL HEADER
WRITE (TAPEOT,260) NCOM
260 FORMAT (1H0,1RHCOMPONENT NUMBER #,13)
WRITE (TAPEOT,270) (COM(I),I=1,NCS)
270 FORMAT (1H0,12HPANEL ID # , 20(A4,2X))
WRITE (TAPEOT,280)
280 FORMAT (1H0,20HSKIN FRICTION FORCES)
WRITE (6,300) MACH,V,REFNO
300 FORMAT (1H0,7H MACH=FR,3,6X VEL=VF9.1,16H FT/SEC RE/FT BE13,5 )
310 FORMAT (1H ,7H ALT =F8,0 )
IF (PSTAG .LT. 0.00001) WRITE (5,310) ALT
IF (PSTAG .GT. 0.00001) WRITE (6,320) PSTAG,YSTAG
320 FORMAT (1H ,16X7HP STAG=F7.1,16M ATMOS T STAG=F7.1,6H DEG F )
WRITE (TAPEOT,330) GTYPE
```

VISC 211
VISC 212
VISC 213
VISC 214
VISC 215
VISC 216
VISC 217
VISC 218
VISC 219
VISC 226C
VISC 2211
VISC 222
VISC 223C
VISC 2241
VISC 225
VISC 226
VISC 227
VISC 228
VISC 229
VISC 230
VISC 231
VISC 232
VISC 233
VISC 234
VISC 235
VISC 236
VISC 237
VISC 238
VISC 239
VISC 240
VISC 241
VISC 242
VISC 243
VISC 244
VISC 245

VISCUS

VISCUS

VISC 246
VISC 247
VISC 248
VISC 249
VISC 250
VISC 251
VISC 252
VISC 253
VISC 254
VISC 255
VISC 256
VISC 257
VISC 258C
VISC 259I
VISC 260
VISC 261
VISC 262
VISC 263
VISC 264
VISC 265
VISC 266
VISC 267
VISC 268
VISC 269
VISC 270C
VISC 271I
VISC 272
VISC 273
VISC 274
VISC 275
VISC 276
VISC 277
VISC 278C
VISC 279I
VISC 280

```
330 FORMAT (1H ,2X,2A4)
WRITE (5,340) SREF,SPAN,MAC,XCG,YCG,ZCG
340 FORMAT (1H0,9H S REF =F9.2,8H SPAN =F8.2,8H MAC =F6.2,1H ,
1 9H X CG =F9.2,8H Y CG =F8.2,8H Z CG =F8.2 )
WRITE (TAPEOT,350)
350 FORMAT (1H0,10MFORCE DATA,1H ,7H ALPHA,9X,3HC 0,7X,3HC L,7X,
1 3HC A,7X,3HC V,7X,3HC N,1H .7H BETA ,9X,3HL/D,7X,3HC M,7X,
2 4HC LL,6X,4HC LN)
C
C WRITE OUTPUT FORCE DATA
DO 370 I=1,NAB
IG9 = LCOM(NTOT) + I*2
CALL READMS (9,P,11,IG9)
READ (9#IG9) P
WRITE (TAPEOT,360) P
360 FORMAT (1H0,P7.2,5F10.5,1H ,F7.2,4F10.5)
370 CONTINUE
C
C
C
C RESET COUNTERS IN GEOMETRY TABLE IF DATA WHERE SAVED
IF (IAS.EQ. 0) GO TO 192
E(1) = NEXTG
IG4 = 2
CALL WRITMS (4,E,25,IG4)
WRITE (4#IG4) E
C
C SAVE COMPONENT INFORMATION IN TABLE OF CONTENTS
192 CONTINUE
DO 191 I=2,11
191 E2(I) = IPANL(I=1)
E2(1) = NTOT
IG9 = LCOM(NTOT)
CALL WRITMS (9,E2,11,IG9)
WRITE (9#IG9) E2
IG9 = IG9 + 1
C
```

VISCUS

VISCUS

VISC 281C
 VISC 282I
 VISC 283
 VISC 284
 VISC 285
 VISC 286
 VISC 287
 VISC 288
 VISC 289C
 VISC 290I
 VISC 291
 VISC 292
 VISC 293
 VISC 294
 VISC 295C
 VISC 296I
 VISC 297
 VISC 298C
 VISC 299I
 VISC 300
 VISC 301
 VISC 302
 VISC 303
 VISC 304
 VISC 305
 VISC 306
 VISC 307
 VISC 308
 VISC 309
 VISC 310
 VISC 311
 VISC 312
 VISC 313
 VISC 314
 VISC 315

CALL WRITMS (9,E4,41,IG9)
 WRITE (9#IG9) E4
 C UPDATE MAIN TABLE OF CONTENTS
 F1(1) = NTOT
 E1(2) = ISUM
 F1(3) = NEXT
 IG9 = 1
 CALL WRITMS (9,E1,8,IG9)
 WRITE (9#IG9) F1
 DO 440 I=2,41
 440 E5(I) = LCOM(I*1)
 F5(1) = 0.0
 IG9 = 2
 CALL WRITMS (9,E5,41,IG9)
 WRITE (9#IG9) E5
 IG9 = 3
 CALL WRITMS (9,E6,41,IG9)
 WRITE (9#IG9) E6
 C
 E8(1) = MACH
 E8(2) = V
 E8(3) = RENO
 E8(4) = ALT
 E8(5) = PSTAG
 E8(6) = TSTAG
 E8(7) = GTYPE(1)
 E8(8) = GTYPE(2)
 E8(9) = SREF
 E8(10) = SPAN
 E8(11) = MAC
 E8(12) = XCG
 E8(13) = YCG
 E8(14) = ZCG
 YG9 = 4

VISCUS

VISCUS

```
CALL WRITMS (9,EB,14,IG9)
WRITE (9#IG9) EB
WRITE (TAPEOT,430) NTOT
430 FORMAT (1H0,10X,35HDATA SAVED ON UNIT 9. 3FT NUMBER #,I4)
CHECK IF LAST COMPONENT HAS BEEN COMPLETED
450 IF (NCOM .LT. NCOMP) GO TO 60
520 GO TO 550
530 WRITE (TAPEOT,540) ERROR,J
540 FORMAT (1H ,10H** ERROR #,I2,26H ON RETURN FROM FORCE. J#,I1)
550 RETURN
550 CONTINUE
END
```

VISC 316C
VISC 317I
VISC 318
VISC 319
VISC 320
VISC 321
VISC 322
VISC 323
VISC 324
VISC 325
VISC 326
VISC 327
VISC 328
VISC 329
VISC 330I
VISC 331C
VISC 332

VISCUS

FORSF

```

SUBROUTINE FORSF (NEXT,IGUS,IPPINT,ISAVE,ISFMTH,LTOTAB)
C
C*****
C*****
C*****
C*****
COMMON /EXEC/CASF, TITLE, PASE, ERRDR
COMMON /GOATA/LTOT, J, SYMFCT, IORN, IGTYPF, L
COMMON /FLIGHT/MACH, ALT, PSTAG, TSTAG, V, RENO, NFB, TFS, AFS, RHOFB, VIS
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 COMMON /ARDATA/NAB, ALPHA(20), BETA(20), RNL(20), COELTA(20), QI(20),
      RI(20), PI(20)
1 COMMON /REF/SPFF, MAC, SPAN, XCG, YCG, ZCG
COMMON /FARRAY/F
COMMON /GASD/GAM, GASC, PRAN, IGAS, AV1, VY2, AY3, GTYPE(2)
COMMON /SKIND/LS, IS(100,9), SURF(100,8) DELTA, SHEAR
COMMON /FSBS/FS(8), BS(8)
DIMENSION F(11), E(25), TITLE(15), E1(25), ANGLE(3)
DIMENSION IGUS(1000)
DIMENSION LTOTAR(20), LCONO(3), DIMFL(6)
INTEGER ERRDR, SYMFCT, SYMFCD, CASE, PAGE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 REAL MACH, MAC, NX, NY, NZ
REAL MACHO, MACHSQ
C
C RAD(A) = A / .572957795E+02
C
C SET UP STARTING CONSTANTS
CA = 0.0
CY = 0.0
CN = 0.0
CLL = 0.0
CLV = 1.0

```

FORS 001
FORS 002
FORS 003
FORS 004
FORS 005
FORS 006
FORS 007
FORS 008
FORS 009
FORS 010
FORS 011
FORS 012
FORS 013
FORS 014
FORS 015
FORS 016
FORS 017
FORS 018
FORS 019
FORS 020
FORS 021
FORS 022
FORS 023
FORS 024
FORS 025
FORS 026
FORS 027
FORS 028
FORS 029
FORS 030
FORS 031
FORS 032
FORS 033
FORS 034
FORS 035

FORSF

FOR8F

FOR8 036
FOR8 037
FOR8 038
FOR8 039
FOR8 040
FOR8 041
FOR8 042
FOR8 043
FOR8 044
FOR8 045
FOR8 046
FOR8 047
FOR8 048
FOR8 049
FOR8 050
FOR8 051
FOR8 052
FOR8 053
FOR8 054
FOR8 055
FOR8 056
FOR8 057
FOR8 058
FOR8 059
FOR8 060
FOR8 061
FOR8 062
FOR8 063
FOR8 064
FOR8 065
FOR8 066
FOR8 067
FOR8 068
FOR8 069
FOR8 076

```
CLN = 0.0  
ARCA = 0.0  
NPCK = 51  
NPRT = 51  
ALPHA = RAD(ALPHA(J))  
BETAR = RAD(BETA(J))  
ROLL = ROL(J)  
PHIP = RAD(ROLL)  
Q = QI(J)  
R = RI(J)  
P = PI(J)  
IDERIV = 0  
SYMFCO = 0  
LTS = LTOT  
IF (ISFMTH ,EQ. 0) LTS = LTOTAB(J)  
  
C INITIALIZE FREE STREAM DATA ARRAY  
FS(1) = RHOPFS  
FS(2) = PFS  
FS(3) = YFS  
FS(4) = AFS  
FS(5) = VIS  
FS(6) = MACH  
FS(7) = MACH * FS(4)  
FS(8) = FS(1)*FS(7)/FS(5)  
G1 = GAM + 1.0  
GM1 = GAM * 1.0  
G4 = 4./G1  
G11 = GM1/G1  
G21 = 2.0*(1.0+G11)  
G2 = 2.0*GAM  
  
C *****START OF ELEMENT DO LOOP *****  
:0 DO 600 L=1,LTY  
IF (ISFMTH ,EQ. 0) GO TO 16
```

FOR8F

FOR9F

```

C GET GEOMETRY DATA FROM NORMAL STORAGE ARRAYS
  IG4 = IG48(L)
  CALL READMS (4,E,25,IG4)
  READ (4#IG4) E
  N = E(2)
  M = E(3)
  NX = E(4)
  NY = E(5)
  NZ = E(6)
  XCENY = E(7)
  YCFNT = E(8)
  ZCFNT = E(9)
  AREA = E(10)

C
C IF (SYMFCO .EQ. 0) GO TO 60
  NY = -NY
  YCFNT = -YCFNT
  GO TO 60

C
C READ GEOMETRY DATA FROM STREAMLINE DATA UNIT
  16 READ(TAPEF) IG4,LCOND,L,DINFL
  C 920 WRITE (TAPEOT,920) IG4,LCOND,E,DINFL
  C 920 FORMAT (1H,4HIG4E,15,3X,6HLCOND=,3I3, /1H,6HE=
  C 1 6X,10E12,4, /1H,6X,5E12,4, /1H,6HDINFL=,6F12,4)
  C
C DUMP GEOMETRY DATA INTO WORKING VARIABLES
  N = E(2)
  M = E(3)
  NX = E(4)
  NY = E(5)
  NZ = E(6)
  XCENY = E(7)
  YCFNT = E(8)
  ZCFNT = E(9)
  AREA = E(10)

```

FOR9 071
FOR9 072
FOR9 073C
FOR9 074I
FOR9 075
FOR9 076
FOR9 077
FOR9 076
FOR9 079
FOR9 080
FOR9 081
FOR9 082
FOR9 083
FOR9 084
FOR9 085
FOR9 086
FOR9 087
FOR9 088
FOR9 089
FOR9 090
FOR9 091
FOR9 092
FOR9 093
FOR9 094
FOR9 095
FOR9 096
FOR9 097
FOR9 098
FOR9 099
FOR9 100
FOR9 101
FOR9 102
FOR9 103
FOR9 104
FOR9 105

FOR9F

FORSF

FORS 106
 FORS 107
 FORS 108
 FORS 109
 FORS 110
 FORS 111
 FORS 112
 FORS 113
 FORS 114
 FORS 115
 FORS 116C
 FORS 117I
 FORS 118
 FORS 119
 FORS 120C
 FORS 121I
 FORS 122
 FORS 123
 FORS 124
 FORS 125
 FORS 126
 FORS 127
 FORS 128
 FORS 129
 FORS 130
 FORS 131
 FORS 132
 FORS 133
 FORS 134
 FORS 135
 FORS 136
 FORS 137
 FORS 138
 FORS 139
 FORS 140

FORSF

```

SHEAR = DIMFL(1)
IG41 = IABS(IG4) + 1
IF (IG4 .LT. 0) IG41 = IG41 + 1
GO TO 70
60 CONTINUE
C OBTAIN SURFACE PROPERTY DATA
IG41 = IG4 + 1
C OBTAIN POINTER TO REFLECTED DATA
IF (SYMFCO .NE. 0) IG41 = IG41 + 1
70 CONTINUE
CALL READMS (4,E1,25,IG41)
READ (4*IG41) E1
C
C
IG4SD = E1(J) + 0.0001
CALL READMS (4,E1,25,IG4SD)
READ (4*IG4SD) E1
C
C
WRITE (TAPEUT,900) IG4SD,E1
C 900 FORMAT (1H ,6HIG4SD=,IS,AE12.5,/1H ,11X,8E12.5,/1H ,11X,9E12.5)
C
C
C CHECK IF DATA MATCHES
LE1 = E1(2) + 0.0001
LE = E (1) + 0.0001
IF (LE1 .EQ. LE) GO TO 250
WRITE (TAPEOT,250)
250 FORMAT (1H0,43H***ELEMENT NUMBER OF SURFACE PROPERTY DATA ,
2 13HPROGRAM STOP. )
WRITE (TAPEOT,251) IG4,J,E
251 FORMAT (1H ,4HIG4=,I4,3H J=,I2,3H E=,8F12.5,/1H ,16X,8E12.5,/1H ,
1 16X,9E12.5)
STOP
C
260 SX = E1(8)
SY = E1(9)
SZ = E1(10)
  
```

FORSF

```

C CHECK SKIN FRICTION METHOD FLAG
C IF (ISFMTH.FG.1) GO TO 300
  CP = 0.0
  DELTA = E1(6)
  GO TO 430
C
C 300 CP = E1(5)
  ANGLE(1) = E1(6)
  DELTA = E1(6)
  RS(6) = E1(7)
  PPINF = E1(11)
  TTINF = E1(12)
  IF (ABS(CP) .GT. 0.00001) GO TO 400
  CO 390 IF 1.8
  390 BS(1) = FS(1)
  GO TO 410
  400 IF (BS(6).EQ.0.0 .OR. PPINF.EQ.0.0 .OR. TTINF.EQ.0.0) GO TO 405
  CALCULATE BS ARRAY USING PPINF AND TTINF OBTAINED FROM SURFACE DATA
C PRESSURE
  BS(2) = PPINF*PFS
C TEMPERATURE
  BS(3) = TTINF * TFS
C DENSITY
  BS(1) = BS(2)/(BS(3)*GASCP*(GM1/GAM))
C SPEED OF SOUND
  BS(4) = SQRT(GASCP*GM1*BS(3))
C VISCOSITY
  IF (RS(3).GE.AV1) BS(5) = 2.27F-8*BS(3)**1.5/(BS(3)+198.6)
  IF (RS(3).LT.AV1) BS(5) = AV2*BS(3)**AV3
  GO TO 408
C
C CALCULATE BS ARRAY GIVEN ONLY CP
C CALCULATE SQUARE OF MACH NUMBER NORMAL TO EFFECTIVE SHOCK
  405 MACHN = FS(6)

```

FORSF

FORF 141
FORF 142
FORF 143
FORF 144
FORF 145
FORF 146
FORF 147
FORF 148
FORF 149
FORF 150
FORF 151
FORF 152
FORF 153
FORF 154
FORF 155
FORF 156
FORF 157
FORF 158
FORF 159
FORF 160
FORF 161
FORF 162
FORF 163
FORF 164
FORF 165
FORF 166
FORF 167
FORF 168
FORF 169
FORF 170
FORF 171
FORF 172
FORF 173
FORF 174
FORF 175

FORSF

```

MACHSQ = (MACHO*MACHO*CP-64)/G21
C CALCULATE EFFECTIVE DENSITY RATIO
EPSI = G11 * (1.0+2.0/(GM1*MACHSQ))
C CALCULATE THE EFFECTIVE SHOCK ANGLE SQUARFD
ANGLE(3) = CP / (2.0*(1.0-EPSI))
C CALCULATE NORMAL MACH SQUARED TIMES SQUARF OF SHOCK ANGLE
EMN = MACHSQ * ANGLE(3)
IF (EMN .LT. 1.01) EMN = 1.01
BS(1) = FS(1)*GM1*EMN/(GM1*EMN+2.0)
C PRESSURE EQ. 128
BS(2) = FS(2)*(G2+EMN*GM1)/G1
C TEMPERATURE EQ. 130
R3 = (G2+EMN*GM1)*(2.0+GM1*EMN)/(G1*GM1*EMN)
BS(3) = FS(3) * R3
C SPEED OF SOUND
BS(4) = FS(4) * SORT(R3)
C VISCOSITY
IF(BS(3).GE.AV1) BS(5) = 2.27E-8*BS(3)**1.5/(BS(3)+198.6)
IF(BS(3).LT.AV1) BS(5) = AV2*BS(3)**.2473
C MACH NUMBER EQ. 132
BS6SQ = ((G1+FS(6))**2*EMN + 4.0*(EMN+1.0))*(GAM+EMN+1.0)/
1 ((2.0+GAM*EMN + GM1)*(2.0+GM1*EMN))
IF (BS6SQ .LT. 1.0) BS(6) = 1.01
IF (BS6SQ .GT. 1.0) BS(6) = SORT(BS6SQ)
C VELOCITY = BS(4) * BS(6)
C REYNOLDS NUMBER PER FOOT
BS(A) = BS(1) * BS(7)/BS(5)
C
C*****
410 LS = L
C CALL THE MARK III SKIN FRICTION ROUTINE
IGTYPE = 2
IF (IS(L,3) .EQ. 1) IGTYPE = 17
WRITE (TAPEOT,910) BS
C

```

FORS 176
FORS 177
FORS 178
FORS 179
FORS 180
FORS 181
FORS 182
FORS 183
FORS 184
FORS 185
FORS 186
FORS 187
FORS 188
FORS 189
FORS 190
FORS 191
FORS 192
FORS 193
FORS 194
FORS 195
FORS 196
FORS 197
FORS 198
FORS 199
FORS 200
FORS 201
FORS 202
FORS 203
FORS 204
FORS 205
FORS 206
FORS 207
FORS 208
FORS 209
FORS 210

FORSF

FOR9P

```

C 910 FORMAT (1H,3HRS=,8E12,5)
      CALL SKINFR
      IF (SURF(L,1) .GT. 0.00001) AREA = SJRF(L,1)
      CP = 0.0
      IF (IGTYPE .EQ. 17) CP = SHEAR / (GAM/2.0*MACH*MACH)
430 SHEARX = SHEAR*SX
      SHEARY = SHEAR*SY
      SHEARZ = SHEAR*SZ
C
440 CONTINUE
C SELECT FORCE METHOD TO MEET SYMMETRY REQUIREMENTS
      IF (SYMFACT .EQ. 1) GO TO 450
      IF (P.NE.0.0 .OR. P.NE.0.0) GO TO 450
      IF (BETA(J).EQ.0.0 .AND. ROLL.EQ.0.0) GO TO 470
C
C CALCULATE SIX-COMPONENT FORCE CONTRIBUTIONS OF ELEMENT
450 DELCA = NX * (CP*AREA/SREF)
      DELCY = NY * (CP*AREA/SREF)
      DELCN = NZ * (CP*AREA/SREF)
      IF (IGTYPE .EQ. 17) GO TO 460
      DELCA = DELCA + SHEARX *(AREA/SREF)
      DELCY = DELCY + SHEARY *(AREA/SREF)
      DELCN = DELCN + SHEARZ *(AREA/SREF)
460 DELCLL = DELCY * (ZCENT - ZCG)/SPAN
      1 DELCLM = DELCN * (YCENT - YCG)/SPAN
      1 DELCLN = DELCN * (XCENT - XCG)/MAC
      1 DELCLN = DELCY * (ZCENT - ZCG)/MAC
      1 DELCLN = DELCY * (XCENT - XCG)/SPAN
      1 DELCLN = DELCA * (YCENT - YCG)/SPAN
      GO TO 500
470 DELCA = NX * (CP*AREA/SREF) + 2.0
      DELCY = 0.0
      DELCN = NZ * (CP*AREA/SREF) + 2.0
      IF (IGTYPE .EQ. 17) GO TO 490
      DELCA = DELCA + SHEARX * 2.0*(AREA/SREF)

```

FOR9 211
FOR9 212
FOR9 213
FOR9 214
FOR9 215
FOR9 216
FOR9 217
FOR9 218
FOR9 219
FOR9 220
FOR9 221
FOR9 222
FOR9 223
FOR9 224
FOR9 225
FOR9 226
FOR9 227
FOR9 228
FOR9 229
FOR9 230
FOR9 231
FOR9 232
FOR9 233
FOR9 234
FOR9 235
FOR9 236
FOR9 237
FOR9 238
FOR9 239
FOR9 240
FOR9 241
FOR9 242
FOR9 243
FOR9 244
FOR9 245

FOR9P

FORSF

FOR3 246
 FOR3 247
 FOR3 248
 FOR3 249
 FOR3 250
 FOR3 251
 FOR3 252
 FOR3 253
 FOR3 254
 FOR3 255
 FOR3 256
 FOR3 257
 FOR3 258
 FOR3 259
 FOR3 260
 FOR3 261
 FOR3 262
 FOR3 263
 FOR3 264
 FOR3 265
 FOR3 266
 FOR3 267
 FOR3 268
 FOR3 269
 FOR3 270
 FOR3 271
 FOR3 272
 FOR3 273
 FOR3 274
 FOR3 275
 FOR3 276
 FOR3 277
 FOR3 278
 FOR3 279
 FOR3 280

FORSF

DELGN = DELCN * SHEARZ * 2.0*(AREA/SREF)

490 DELCLL = 0.0

DELCLM=(DELGN * (XCENT * XCG) / MAC

1 * DELCA * (ZCENT * ZCG) / MAC)

DELCLN = 0.0

C SUM UP SIX-COMPONENT FORCE CONTRIBUTIONS

500 CA = CA + DELCA

 CY = CY + DELCY

 CN = CN + DELCN

 CLL = CLL + DELCLL

 CLM = CLM + DELCLM

 CLN = CLN + DELCLN

 AREAT = AREAT + AREA

510 IF (IPRINT .EQ. 0) GO TO 505

C CHECK TO PRINT HEADER AT TOP OF PAGE

520 IF (NPRT.LT.NPCK) GO TO 580

530 NPRT = 0

 CALL HEADER

540 WRITE (TAPEOT,540) (GTYPE(I),I=1,2)

550 FORMAT (1H , 63' 2A4)

560 CONTINUE

570 WRITE (TAPEOT,560) MACH,ALT,SREF,SPAN,XCG,YCG,ZCG,MAC

580 FORMAT (1H0,20ELEMENT DATA MACH=F7.3,7M ALT =F8.0,9M S REF =

1 F8.1,8M SPAN =F7.1,7M

2 15XSHXCG =F7.1,7M YCG =F7.1,10M ZCG =F7.1,4XSHMAC =F7.1)

WRITE (TAPEOT,570) ALPHA(J),BETA(J),

1 JOERIV,G,R,P

570 FORMAT (1H ,5X17HANGLE OF ATTACK =F6.2,3X11HYAW ANGLE =F6.2,71M ,

2 5XAMIDERIV =I3,3X3HQ =E12.5,3X,3HR =E12.5,3X3MP =E12.5,7

3 1H0,2X,1HL,6X,6MDEL CA,8X

4 6MDEL CY,6X6MDEL CN,7X7MDEL CLL,7X7MDEL CLM,7X7MDEL CLN,7X2HCP,

5 13X4MAREA,71M ,11X2HCA,12X2HCY,12X2HCN,11X3MCLL,11X3MCLM,

6 12X3MCLN,6X5MDELTA/1M ,10X5KXCENT, 9X5HZCENT)

NPRT = NPRT + 15

FORSF

```
C PRINT ELEMENT DATA
C 580 WRITE(TAPEOT,590) L,DELCA,DEL CY,DEL CN,DEL CL,DEL CLM,DEL CLN,CP,AREA,
1 CA,CY,CN,CLL,CLM,CLN,DELTA,XCENT,YCENT,ZCENT
C 590 FORMAT (I10,I4,8E14.5,I1H,4X7E14.5,I1H,4X3E14.5)
      NPRT = NPRT + 4
C SET UP AND SAVE ELEMNT RESULTS ON GEOMETRY STORAGE UNIT
C 595 IF (ISAVE .EQ. 0) GO TO 600
      E1(3) = SHEAR
      CALL WRITMS (4,E1,25,IG4SD)
      WRITE (4#IG4SD) E1
C
C *****
C 600 CONTINUE
C ***** END OF DO LOOP FOR ALL ELEMENTS
C
C
C CHECK IF NO SYMMETRY
      IF (ISEMTH .EQ. 0) GO TO 620
      IF (SYMFCT .EQ. 1) GO TO 620
C CHECK IF YAW AND ROLL ARE ZERO
      IF (BETA(J).EQ.0.0 .AND. ROLL.EQ.0.0) GO TO 620
C CHECK IF SYMMETRY CALCULATIONS ARE COMPLETED
      IF (SYMFCD .EQ. 1) GO TO 620
C SET SYMMETRY RE-CYCLE FLAG
      SYMFCD = 1
      GO TO 10
C 620 CONTINUE
C SET UP ARRAY OF FINAL DATA
      F(1) = ALPHA(J)
      F(4) = CA
      F(5) = CY
      F(6) = CN
      F(7) = BETA(J)
      F(9) = CLM
```

FORS 281
FORS 282
FORS 283
FORS 284
FORS 285
FORS 286
FORS 287
FORS 288
FORS 289
FORS 290C
FORS 291I
FORS 292
FORS 293
FORS 294
FORS 295
FORS 296
FORS 297
FORS 298
FORS 299
FORS 306
FORS 301
FORS 302
FORS 303
FORS 304
FORS 305
FORS 306
FORS 307
FORS 308
FORS 309
FORS 310
FORS 311
FORS 312
FORS 313
FORS 314
FORS 315

FORSF

FORSF

```
F(10) = CLL
F(11) = CLN
C RESOLVE NORMAL AND AXIAL FORCES IN LIFT AND DRAG DIRECTION
CD = CA * COS(ALPHAR) * COS(BETAR) - CY * SIN(PHIR) * SIN(ALPHAR) * COS(BETAR)
1 -CY * COS(PHIR) * SIN(BETAR) + CN * COS(PHIR) * SIN(ALPHAR) * COS(BETAR)
2 -CN * SIN(PHIR) * SIN(BETAR)
CL = -CA * SIN(ALPHAR) * CY * SIN(PHIR) + COS(ALPHAR) * CN * COS(PHIR)
1 * COS(ALPHAR)
C CALCULATE SIDE FORCE COEFFICIENT = WIND AXIS
CYPRIM = CA * COS(ALPHAR) * SIN(BETAR) - CY * SIN(PHIR) * SIN(ALPHAR)
1 * SIN(BETAR) + CY * COS(PHIR) * COS(BETAR) + CN * COS(PHIR) * SIN(ALPHAR)
2 * SIN(BETAR) + CN * SIN(PHIR) * COS(BETAR)
C
C SET UP REMAINING PARAMETERS
F(2) = CD
F(3) = CL
IF (CD .EQ. 0.0) CD = 0.000001
F(6) = CL / CD
C
C 630 RETURN
END
```

```
FORS 316
FORS 317
FORS 318
FORS 319
FORS 320
FORS 321
FORS 322
FORS 323
FORS 324
FORS 325
FORS 326
FORS 327
FORS 328
FORS 329
FORS 330
FORS 331
FORS 332
FORS 333
FORS 334
FORS 335
FORS 336
FORS 337
```

FORSF

SKINFR

SKIN 036
 SKIN 037
 SKIN 038
 SKIN 039
 SKIN 040
 SKIN 041
 SKIN 042
 SKIN 043
 SKIN 044
 SKIN 045
 SKIN 046
 SKIN 047
 SKIN 048
 SKIN 049
 SKIN 050
 SKIN 051
 SKIN 052
 SKIN 053
 SKIN 054
 SKIN 055
 SKIN 056
 SKIN 057
 SKIN 058
 SKIN 059
 SKIN 060
 SKIN 061
 SKIN 062
 SKIN 063
 SKIN 064
 SKIN 065
 SKIN 066
 SKIN 067
 SKIN 068
 SKIN 069
 SKIN 070

10 IF (MACH .GT. 1.0) GO TO 30
 WRITE (TAPEOT,20)
 20 FORMAT (1H0,87H*** INPUT MACH NUMBER IS NOT SUPERSONIC, SKIN
 1 49H FRICTION ANALYSIS FOR THIS POINT IS STOPPED **** ;
 SKIN = 0.0
 GO TO 510

C SET STARTING CONSTANTS
 30 MFR = 1
 SCFA(1) = 0.0
 SCFA(2) = 0.0
 IF (LS .EQ. 1) NPRT = 0
 IGTY = 0
 IF (IGTYPE .EQ. 17) IGTY = 1
 IF (IGTYPE .EQ. 17) IGTYP = 2
 MEREXP = 0
 NC = 0
 NS = LS
 SCF(1) = 0.0
 SCF(2) = 0.0
 SCF(3) = 0.0
 SCF(4) = 0.0

C VISCOSITY EQUATION
 GAB GAM
 TR(1) = ALT
 TR(2) = MACH
 TR(3) = MACH * FS(4)
 TR(4) = 0.0
 ANGLE(1) = DELTA
 I = LS

C
 C
 C
 C DETERMINE APPROPRIATE LENGTHS AND TAPER RATIOS.

SKINFR

SKINPR

SKIN 071
 SKIN 072
 SKIN 073
 SKIN 074
 SKIN 075
 SKIN 076
 SKIN 077
 SKIN 078
 SKIN 079
 SKIN 080
 SKIN 081
 SKIN 082
 SKIN 083
 SKIN 084
 SKIN 085
 SKIN 086
 SKIN 087
 SKIN 088
 SKIN 089
 SKIN 090
 SKIN 091
 SKIN 092
 SKIN 093
 SKIN 094
 SKIN 095
 SKIN 096
 SKIN 097
 SKIN 098
 SKIN 099
 SKIN 100
 SKIN 101
 SKIN 102
 SKIN 103
 SKIN 104
 SKIN 105

```

EL = SURF(I,2)
ELO = SURF(I,3)
TAPER1 = SURF(I,4)
TAPER2 = SURF(I,5)
IF (TAPER1 .LT. 0.0) GO TO 40
SURF(I,5) = (EL*TAPER2 + ELO*TAPER1)/(FL + ELO)
EL = EL + ELO
GO TO 70
40 TAPER1 = -TAPER1
SURF(I,5) = 1.0
EL1 = ELO + EL*TAPER2
FL = EL + ELO*TAPER1
IF (FL-EL1)50,70,60
50 SURF(I,5) = EL/EL1
FL = EL1
GO TO 70
60 SURF(I,5) = FL1/EL
70 IF (TAPER1 .EQ. 0.0) TAPER1 = 0.0001
IF (SURF(I,5) .EQ. 0.0) SURF(I,5) = 0.0001
TR(5) = SURF(I,6)
TR(6) = SURF(I,7)
C
IF (SURF(I,5) .LT. 0.8) GO TO 80
ELL = EL + 4.0 * ((1.0 + SURF(I,5)) / (3.0+SURF(I,5)))**2
GO TO 90
C
80 ELL = EL + (0.75*(1.0-SURF(I,5)**2) / (1.0-SURF(I,5)))**2
90 REYN = FS(8) * ELL
C
C CALCULATE REFERENCE TEMPERATURE, REYNOLDS NO., AND WALL
C TEMPERATURE (IF REQUIRED). FIRST CHECK IF TEMPERATURE
C ITERATIONS AND/OR LOCAL CF DATA TO BE PRINTED.
100 IF (IS(I,9).EQ.0) GO TO 120
C
C LOCAL DATA TO BE PRINTED. IF TEMPERATURE ITERATIONS START
  
```

SKINPR

SKINFR

```
C NEW PAGE FOR EACH SURFACE. OTHERWISE TEST IF HEADER REQUIRED.
  IF (IS(I,9).EQ.1) GO TO 110
  IF ((NPR7.LT.50).AND.(LS.NE.1)) GO TO 120

C
  110 CALL HEADER
      WRITE (TAPEOT,350) (GTYPE(IK),IK=1,2)
      WRITE (TAPEOT,360) ALPHA(J),MACH,TR(3),ALT,FS(8),SREF
      NPR7 = 9
  120 CALL TEMPEL,TR,RE,TS,IS(I,6),MER,IS(I,9),RT,CF)
  C CHECK ERROR FLAG
      IF (MER .LE. 1) GO TO 130
      MER = 1
      GO TO 330

C
  C CALCULATE VISCOUS=INVISCID INTERACTION EFFECT ON SKIN FRICTION
  130 KO = MACH * SIN(ANGLE(1) / 57.29578)
      IF (KO.LT..0.0001 .OR. KO.GT.0.0001) GO TO 140
      PO = 1.0
      M = 1.9156
      GO TO 170
  140 CONTINUE
      IF (KO.GE.0.0) GO TO 150
  C EXPANSION SURFACE.
      PO = 0.00001
      M = 0.00001
      IF (KO.LE.(=2./(GA*1.))) GO TO 170
      PO = (1.0 + 0.5*(GA-1.))*KO)**(2.*GA/(GA-1.))
      IF (KO.GE..3.0) GO TO 160
      M = 1.424 + 0.219*KO
      GO TO 170
  C COMPRESSION SURFACE.
  150 PO = 1.0 + 0.25*GA*(GA + 1.0)*KO*KO +
      GA*KO*SQRT(1.0 + (0.25*(GA + 1.0)*KO)**2)
  160 M = 1.9156 + KO*(0.41727 + KO*(0.0419101 + KO*(0.010427
      + KO*(0.00214381 + KO*1.03217E-4))))
      1
```

SKIN 106
SKIN 107
SKIN 108
SKIN 109
SKIN 110
SKIN 111
SKIN 112
SKIN 113
SKIN 114
SKIN 115
SKIN 116
SKIN 117
SKIN 118
SKIN 119
SKIN 120
SKIN 121
SKIN 122
SKIN 123
SKIN 124
SKIN 125
SKIN 126
SKIN 127
SKIN 128
SKIN 129
SKIN 130
SKIN 131
SKIN 132
SKIN 133
SKIN 134
SKIN 135
SKIN 136
SKIN 137
SKIN 138
SKIN 139
SKIN 140

SKINFR

SKINFR

```

IF (KO .GT. 10.0) M = SQRT(2.0*GA*(GA+1.0))
170 TWTRL = TR(5) / RT(1)
      A = 0.8604*(GA+1.)*TWTRL+(4.65*PRAN**(-.166667)+3.65)*PRAN/2.59)
      A = G / 2.0
IF (TR(5).GE.AV1) VISWL = 2.27E-8*TR(5)**1.5/(TR(5)+198.6)
IF (TR(6).GE.AV1) VISWT = 2.27E-8*TR(6)**1.5/(TR(6)+198.6)
IF (TR(5).LT.AV1) VISWL = AV2*TR(5)**AV3
IF (TR(6).LT.AV1) VISWT = AV2*TR(6)**AV3
RELOC = RS(8) * SURF(I,2)
CLAM = VISWL * TFS / (-IS*TR(5))
CTURB = VISWT * TFS / (VIS*TR(6))
CHIBAR = MACH**3*SQRT(CLAM) / SQRT(RENM*EL/ELL)
VBAR = CHIBAR / (MACH*MACH)
IF (TS(1).GE.AV1) VISTAR = 2.27E-8*TS(1)**1.5/(TS(1)+198.6)
IF (TS(1).LT.AV1) VISTAR = AV2*TS(1)**AV3
CSTAR = VISTAR * TFS / (VIS*TS(1))
VSTAR = MACH * SQRT(CSTAR) / SQRT(RENM*EL/ELL)
FJ = IS(I,2)
LAMBDA = A * CHIBAR / SQRT(1. + 2.*FJ)
IF (IGTY .EQ. 1) GO TO 500
B = M * LAMBDA / PG * SQRT(EL/ELL)
CFCFOL = SQRT(1.0+8) + 0.5*8*ALOG(ABS((SQRT(1.0+8)+1.0)/
1      (SQRT(1.0+8)-1.0)))
      IF (CFCFOL .LT. 1.0) CFCFOL = 1.0
      CFCFOT = 1.0

C   START TURBULENT FLOW CALCULATIONS
      IF (RE(2) .LE. 6570.) GO TO 260
      EN = 0.8686 / (ALOG10(RE(2)) + 2.0)
C   CHECK TAPER RATIO AND CHARACTERISTIC LENGTH TERMS
      IF (SURF(I,5) .LT. 0.8) GO TO 180
      O = SQRT ((1.0 + SURF(I,5)) / (1.0 + EN + SURF(I,5) * (1.0-EN)))
      GO TO 190
C

```

SKIN 141
SKIN 142
SKIN 143
SKIN 144
SKIN 145
SKIN 146
SKIN 147
SKIN 148
SKIN 149
SKIN 150
SKIN 151
SKIN 152
SKIN 153
SKIN 154
SKIN 155
SKIN 156
SKIN 157
SKIN 158
SKIN 159
SKIN 160
SKIN 161
SKIN 162
SKIN 163
SKIN 164
SKIN 165
SKIN 166
SKIN 167
SKIN 168
SKIN 169
SKIN 170
SKIN 171
SKIN 172
SKIN 173
SKIN 174
SKIN 175

SKINFR

SKZMFR

SKIN	176
SKIN	177
SKIN	178
SKIN	179
SKIN	180
SKIN	181
SKIN	182
SKIN	183
SKIN	184
SKIN	185
SKIN	186
SKIN	187
SKIN	188
SKIN	189
SKIN	190
SKIN	191
SKIN	192
SKIN	193
SKIN	194
SKIN	195
SKIN	196
SKIN	197
SKIN	198
SKIN	199
SKIN	200
SKIN	201
SKIN	202
SKIN	203
SKIN	204
SKIN	205
SKIN	206
SKIN	207
SKIN	208
SKIN	209
SKIN	210

SKINFR

```

180 Q = SORT ((1.0 - SURF(I,5)**2) *(1.0 - 0.5 *EN) /
1 (1.0 - SURF(I,5)**(2.0-EN)))
190 ELT = EL * (RE(2)/10.0**1.5) ** (Q * 1.0)
ESIN = SIN((ANGLE(1)-TR(4))/57.29578)
ECOS = COS((ANGLE(1)-TR(4))/57.29578)
RE(2) = RE(2) * ELT/EL
CFT(1) = 0.088 / (0.43429448 * ALOG(RE(2)) * 1.5)**2
FF = ELO /SURF(I,2)*(1.0 + TAPER1)/(1.0 + TAPER2)
IF (ELO.LT. 0.0001) GO TO 250
IF ((RE(2)*ELO/ELT).GT. 6570.0) GO TO 220
IF (TAPER1.LT. 0.8) GO TO 200
EL1 = ELO*4.0*((1.+TAPER1)/(3.0+TAPER1))**2
GO TO 210
200 EL1 = ELO*(0.75*(1.+TAPER1**2)/(1.+TAPER1**1.5))**2
210 CFL(1) = 1.328/SORT(RE(2)*EL1/ELT)
FF = FF*(CFL(1)/CFT(1) * 1.0)
GO TO 250
220 EN = 0.8686/(ALOG10(RE(2)*ELO/ELT) - 2.0)
IF (TAPER1.LT. 0.8) GO TO 230
Q = SORT((1.0+TAPER1)/(1.0+EN+TAPER1*(1.-EN)))
GO TO 240
230 Q = SORT((1.0+TAPER1**2)*(1.0+0.5*EN)/(1.0+TAPER1**2.0+EN))
240 EL1 = ELO*(RE(2)*ELO/ELT/10.**1.5)**(Q-1,0)
FF = FF*((ALOG10(RE(2)))-1.5)/(ALOG10(RE(2)*EL1/ELT)
1 -1.5)**2 * 1.0)
250 CFT(1) = CFT(1)*(1.0+FF)
CFT(2) = CFT(1) /FC
CFT(3) = CFT(2) * (BS(7)/FS(7))**2 * BS(1) / FS(1) * CFCFOY
CFT(4) = CFT(3) * SURF(I,1) / SREF
IF (IGTYPE.EQ. 2) GO TO 260
IF (IS(I,5).EQ.0) CFT(5) = -ESIN * CFT(4)
IF (IS(I,5).EQ.0) CFT(6) = ECOS * CFT(4)
IF (IS(I,5).EQ.1) CFT(5) = 0.0
IF (IS(I,5).EQ.1) CFT(6) = CFT(4) * COS(SURF(I,4)/57.29578)

```

SKINFR

SKIN	211
SKIN	212
SKIN	213
SKIN	214
SKIN	215
SKIN	216
SKIN	217
SKIN	218
SKIN	219
SKIN	220
SKIN	221
SKIN	222
SKIN	223
SKIN	224
SKIN	225
SKIN	226
SKIN	227
SKIN	228
SKIN	229
SKIN	230
SKIN	231
SKIN	232
SKIN	233
SKIN	234
SKIN	235
SKIN	236
SKIN	237
SKIN	238
SKIN	239
SKIN	240
SKIN	241
SKIN	242
SKIN	243
SKIN	244
SKIN	245

SKINFR

```

C START LAMINAR FLOW CALCULATIONS
260 RE(1) = RE(1) * ELL/EL
    ESIN = SIN((ANGLE(1)-TR(4)))/57.29578)
    FCOS = COS((ANGLE(1)-TR(4)))/57.29578)
    FF = FLO /SURF(I,2)*(1.0 + TAPER1)/(1.0 + TAPER2)
    IF (ELO.LT. 0.0001) GO TO 290
    IF (TAPER1.LT. 0.8) GO TO 270
    EL1 = FLO*4.0*((1.0 + TAPER1)/(3.0 + TAPER1))**2
    GO TO 280
270 EL1 = ELO*(0.75*(1.0-TAPER1**2)/(1.0-TAPER1**1.5))**2
280 R = R*SQRT(ELL/EL1)
    CFF(1) = SQRT(1.0+3) + 0.5*B*ALOG(ABS((SQRT(1.0+B)+1.0)/
    (SQRT(1.0+B)-1.0)))
1 FF = FF*(SQRT(ELL/FL1)*CFCF(1)/CFCFOL * 1.0)
290 CFL(1) = 1.328/SQRT(RE(1))*(1.0-FF)
    CFL(2) = CFL(1) * BS(3)/TS(1)
    CFL(3) = CFL(2) * (BS(7)/FS(7))**2 * BS(1) / FS(1) * CFCFOL
    CFI(4) = CFL(3) * SURF(I,1) / SKEF
    IF (IGTYPE .EQ. 2) GO TO 310
    IF (IS(I,5).EQ.0) CFL(5) = -ESIN * CFL(4)
    IF (IS(I,5).EQ.0) CFL(6) = FCOS * CFL(4)
    IF (IS(I,5).EQ.1) CFL(5) = 0.0
    IF (IS(I,5).EQ.1) CFL(6) = COS(SURF(I,4)/57.29578)
C CALCULATE TOTALS USING LAMINAR FLOW
    SCF(1) = SCF(1) + CFL(5)
    SCF(2) = SCF(2) + CFL(6)
C TOTAL SKIN FRICTION FORCE COEFF. AND SURFACE
    SCFA(1) = SCFA(1) + CFL(4)
C
C CHECK IF RE(2) IS LOWER THAN THE CUTOFF POINT (USE LAMINAR PROP.)
    IF (RE(2) .GT. 6570.0) GO TO 300
    SCF(3) = SCF(3) + CFL(5)
    SCF(4) = SCF(4) + CFL(6)

```

SKINPR

SKIN	246
SKIN	247
SKIN	248
SKIN	249
SKIN	250
SKIN	251
SKIN	252
SKIN	253
SKIN	254
SKIN	255
SKIN	256
SKIN	257
SKIN	258
SKIN	259
SKIN	260
SKIN	261
SKIN	262
SKIN	263
SKIN	264
SKIN	265
SKIN	266
SKIN	267
SKIN	268
SKIN	269
SKIN	270
SKIN	271
SKIN	272
SKIN	273
SKIN	274
SKIN	275
SKIN	276
SKIN	277
SKIN	278
SKIN	279
SKIN	280

SKINPR

```

SCFA(2) = SCFA(2) + CFL(4)
CFT(3) = CFL(3)
CFT(5) = CFL(5)
CFT(6) = CFL(6)
GO TO 320
C CALCULATE TOTALS USING TURBULENT FLOW
300 SCF(3) = SCF(3) + CFT(5)
SCF(4) = SCF(4) + CFT(6)
C TOTAL SKIN FRICTION FORCE COEFF. AND SURFACE
SCFA(2) = SCFA(2) + CFT(4)
GO TO 320
310 CFL(6) = 0.0
CFL(5) = 0.0
SCF(2) = 0.0
SCF(1) = 0.0
CDL = 0.0
CFT(6) = 0.0
CFT(5) = 0.0
SCF(4) = 0.0
SCF(3) = 0.0
CDT = 0.0
IF (RE(2) .LE. 6570.0) CFT(3) = CFL(3)
C
C CALCULATE RE#/FT
320 RE1 = RE(1) / ELL
RE2 = RE(2) / ELL
RE3 = RE(2) / ELL
TWTL = TR(5) / FS(3)
TWTT = TR(6) / FS(3)
TWTRL = TR(5) / RT(1)
TWTRT = TR(6) / RT(2)
IF (IGTYPE .EQ. 2) GO TO 330
ALPHAR = ALPHA(J)/57.295779
CDT = CFT(5)*SIN(ALPHAR) + CFT(6)*COS(ALPHAR)
CDL = CFL(5)*SIN(ALPHAR) + CFL(6)*COS(ALPHAR)

```

SKINFR

```
C
C CHECK IF DATA IS TO BE PRINTED
330 IF (IS(I,7).EQ.0 .AND. MREXP.EQ.0) GO TO 490
    IF (IS(I,7).EQ.0 .AND. MEREXP.NE.0) GO TO 470
C PRINT SKIN FRICTION DATA
C
    IF (IS(I,9).EQ.0) GO TO 340
    IF ((NS.EQ.1).AND.(MREXP.FG.1)) GO TO 340
    IF (IS(I,9).FG.1) GO TO 370
    IF ((NS.NE.1).AND.(NPRT.GT.10)) GO TO 400
    GO TO 370
C
C CHECK IF HEADER IS REQUIRED
340 IF ((NS.NE.1).AND.(NPRT.LT.50)) GO TO 400
    CALL HEADER
    WRITE (TAPEOT,350) (GTYPE(IK),IK=1,2)
350 FORMAT (1H , 63X, 2A4)
    NPRT = 5
C
    WRITE (TAPEOT,360) ALPHA(J),MACH,TR(3),ALT,FS(6),SREF
360 FORMAT (1H0,
1 23H FREE STREAM CONDITIONS,/1H ,9X7HAI PHA =F7.2,6X9HMACH =
2 F6.2,3X10HVELOCITY =F9.1,3X10HALTITUDE =F9.1,/1H ,9X7HRE/FT =
3 1PE10.3,3X9HS REF =0PF9.1)
    NPRT = NPRT + 4
C
370 WRITE (TAPEOT,380)
    NPRT = NPRT + 5
380 FORMAT (1H0,19H SKIN FRICTION DATA,/1H ,9X6HSURF NO.,3X4HTYPE,
1 4X6HMETHOD,4X5HS WET,3X7HLENGTHL,2X7HLENGTHI,3X5HTAPER,
2 4X6HANGLE(1),2X6HRE LOC,4X7HCHI BAR,2X5HV BAR,/
3 1H ,2X3HLAM,7X2HCF,7X2HCA,7X2HCN,5X6HSUM CA,3X6HSUM CN,5X2HTW,
4 7X4HTW/T,6X5HTW/TR,3X6HRE*/FT,4X7H CD ,2X6HCF/CFO,/
5 1H ,2X4HTURB,6X2HCF,7X2HCA,7X2HCN,5X6HSUM CA,3X6HSUM CN,5X2HTW,
6 7X4HTW/T,6X5HTW/TR,3X6HRE*/FT,4X7H CD ,2X6HCF/CFO )
```

SKIN 281
SKIN 282
SKIN 283
SKIN 284
SKIN 285
SKIN 286
SKIN 287
SKIN 288
SKIN 289
SKIN 290
SKIN 291
SKIN 292
SKIN 293
SKIN 294
SKIN 295
SKIN 296
SKIN 297
SKIN 298
SKIN 299
SKIN 300
SKIN 301
SKIN 302
SKIN 303
SKIN 304
SKIN 305
SKIN 306
SKIN 307
SKIN 308
SKIN 309
SKIN 310
SKIN 311
SKIN 312
SKIN 313
SKIN 314
SKIN 315

SKINFR

SKINFR

SKIN 316
 SKIN 317
 SKIN 318
 SKIN 319
 SKIN 320
 SKIN 321
 SKIN 322
 SKIN 323
 SKIN 324
 SKIN 325
 SKIN 326
 SKIN 327
 SKIN 328
 SKIN 329
 SKIN 330
 SKIN 331
 SKIN 332
 SKIN 333
 SKIN 334
 SKIN 335
 SKIN 336
 SKIN 337
 SKIN 338
 SKIN 339
 SKIN 340
 SKIN 341
 SKIN 342
 SKIN 343
 SKIN 344
 SKIN 345
 SKIN 346
 SKIN 347
 SKIN 348
 SKIN 349
 SKIN 350

SKINFR

```

C      IF (IS(I,8) .EQ. 0) GO TO 400
      WRITE (TAPEOT,390)
      NPRT = NPRT + 1
390  FORMAT (1M,10X4MMACH,8X1HV,5X7HV SUJUN,3X5HP=PPF,3X6HTEMP=R,
      1 3X9HRHD*10**4,1X9HVIS*10**7,3X5HRE/FT,2X,6HC STAR,7X,1HC,5X,
      2 6HV STAR)
C      400 WRITE (TAPEOT,410) (IS(I,K),K=1,3), (SURF(I,K),K=1,4), ANGLE(1),RELO
      1C      ,CHIBAR,VBAR
      NPRT = NPRT + 2
410  FORMAT (1M0,11X12,9XI:1,7X12,4XF8,0,2XF4,1,4XF5,1,3XF5,1,6XF6,2,
      1 1X1PE10,3,2XOPF7,3,1XF7,4 )
C      IF (MEREXP=NE,0) GO TO 450
C      WRITE (TAPEOT,420) CFL(3),CFL(6),CFL(5),SCF(2),SCF(1),TR(5),TWTL,
      1  TWTRL,RE1,CDL      ,CFCL,
      2  CFT(3),CFT(6),CFT(5),SCF(4),SCF(3),TR(6),TWTT,
      3  TWTRT,RE2,CDT      ,CFCLD
      NPRT = NPRT + 2
420  FORMAT (1M ,2X3MLAM,4XF8,5,F9,5,F9,5,F9,5,F9,5,F9,5,F9,1,F10,4,
      1 F10,4,1PE10,5,OPF9,5,1XF7,4,
      2 1M ,2X4MTURB,3XF8,5,F9,5,F9,5,F9,5,F9,5,F9,1,F10,4,F10,4,
      3 1PE10,3,OPF9,5,1XF7,4 )
C      430 IF (IS(I,8) .EQ. 0) GO TO 480
      WRITE (TAPEOT,440) FS(6),FS(7),FS(4),FS(2),FS(3),FS(1),FS(5),FS(8)
      1, CSTAR,CLAM,VSTAR,BS(6),BS(7),BS(4),BS(2),BS(3),BS(1),BS(5),BS(8)
      NPRT = NPRT + 2
440  FORMAT (1M ,2X6HSTREAM,F9,5,F9,1,F9,2,F9,4,F9,2,4PF10,7,7PF10,6,
      11PE10,3,2E10,3,OPF7,4,1M ,2X6HLOCAL ,7PF9,5,F9,1,F9,2,F9,4,
      2  F9,2,4PF10,7,7PF10,6,1PE10,3)
      GO TO 480
C

```

SKINPR

```

450 WRITE (TAPEOT,440)
      NPRT = NPRT + 1
460 FORMAT (1H ,10X43MFLOW EXPANDED TO A VACUUM = SURFACE DELETED )
470 MREXP = 0
      CFL(3) = 0.0
      CFT(3) = 0.0
      IF (IS(I,7) .NE. 0) GO TO 410
C   STEP PRINT FLAG
480 CONTINUE
C
C   490 IJ = I
C
C
C   ADD SKIN FRICTION DRAG TO AXIAL FORCE COEFFICIENT
      IF (IS(IJ,4) .EQ. 0) SKIN = CFT(3)
      IF (IS(IJ,4) .NE. 0) SKIN = CFL(3)
      GO TO 510
C
C   CALCULATE INDUCED PRESSURE INCREMENT
500 SKIN = 8./3.*M*LAMBDA*FL/SURF(I,2)*((1.+SURF(I,5)+SQRT(SURF(I,5)))
      1 /((1.+SQRT(SURF(I,5)))+(1.+TAPER2)))+(1.+TAPER1)+SQRT(TAPER1))/
      2 ((1.+SQRT(TAPER1))+(1.+TAPER2))*SQRT(FLD/EL)
C
510 RETURN
C
      END
```

SKIN 351
SKIN 352
SKIN 353
SKIN 354
SKIN 355
SKIN 356
SKIN 357
SKIN 358
SKIN 359
SKIN 360
SKIN 361
SKIN 362
SKIN 363
SKIN 364
SKIN 365
SKIN 366
SKIN 367
SKIN 368
SKIN 369
SKIN 370
SKIN 371
SKIN 372
SKIN 373
SKIN 374
SKIN 375
SKIN 376

SKINPR

TEMP

TEMP	001
TEMP	002
TEMP	003
TEMP	004
TEMP	005
TEMP	006
TEMP	007
TEMP	008
TEMP	009
TEMP	010
TEMP	011
TEMP	012
TEMP	013
TEMP	014
TEMP	015
TEMP	016
TEMP	017
TEMP	018
TEMP	019
TEMP	020
TEMP	021
TEMP	022
TEMP	023
TEMP	024
TEMP	025
TEMP	026
TEMP	027
TEMP	028
TEMP	029
TEMP	030
TEMP	031
TEMP	032
TEMP	033
TEMP	034
TEMP	035

TEMP

SUBROUTINE TEMP(EL,TR,RE,TS,NM,NER,IPRINT,RT,CF)

C
C CALCULATES WALL TEMPERATURE AND OTHER QUANTITIES NECESSARY
C FOR SKIN FRICTION AND BOUNDARY LAYER CALCULATIONS.
C NM IS CONTROL FLAG WITH OPTIONS GIVEN BY FOLLOWING MATRIX.

GAS TYPE	EQUIL	WALL TEMP.	TYPE	INPUT	LAMINAR	METHOD FOR TURBULENT
IDEAL	0	1	2	REF. T	S=C	
REAL	3	4	5	REF. M	S=C	
IDEAL	6	---	7	REF. T	REF. T	
REAL	8	---	9	REF. M	REF. M	

C BOTH LAMINAR (K = 1) AND TURBULENT (K = 2) DETERMINED.
C*****D. N. SMYTH PROGRAM AUTHOR*****

DIMENSION CF(2), BS(8),FS(6),RE(2),TR(10),TS(2),RT(2),RF(2)
COMMON /SKIND/LS,IS(100,9),SURF(100,6),DELTA,SKIN
COMMON /FSBS/FS,RS
COMMON /TAPE/TAPEIN,TAPEOT,TAPEI,TAPEJ,TAPEK
1 TAPEG,TAPEH,TAPEL,TAPEM,TAPEO,TAPEP,TAPEQ,TAPEE,TAPEF,
TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
1 TAPEL,TAPEM,TAPEO,TAPEP,TAPEQ,TAPEE,TAPEF,
TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /TEMPGC/MAW,H2,M1,HW,CKU,FC,FRX;RET,ELLOC,GCP, TST1,ROMJRA
COMMON /FLAG2/ITW,IHW,IFLOW,ITURB,CPTLJC
COMMON /GASD/GAM,GASC.,PRAN,IGAS,AV1,AV2,AV3,GTYPF(2)

TEMP

```

C INTEGER CASE, PAGE, ERROR
  DATA EPST, KTMX, EMISS/5, F=4, 10, 0.8/
C SET UP GENERAL QUANTITIES (GCP CONSISTENT WITH ATMOS).
  G = GAM
  GCP = GASCP
  M1 = GCP*FS(3)
  HTOT = M1*(1. + 0.5*(G=1.))*FS(6)**2)
  M2 = M1*(1. + 0.5*(G = 1.))*FS(6)**2*(1. + (BS(7)/FS(7))**2))
  IF (EL, LE, 0.0) EL = 0.01
  ELLOC = EL
  RF(1) = SQR7(PRAN)
  RF(2) = PRAN**(1./3.)
  CKU = 0.332*FS(5)*SQR7(FS(8)/EL)/PRAN**(2./3.)*SQR7(BS(7)/FS(7))
  RADK = 0.480F=12*EMISS*779.0

```

```

C SET UP CONTROL FLAGS
  ITURB = 1
  IF (NW.GT.5) ITURB = 2
  NWI = NW
  ITW = 1
  IF (NWI.LT.3) GO TO 30
  ITW = 2
  NWI = NWI * 3
  IF (NWI.LT.3) GO TO 30
  IF (NWI.LT.5) GO TO 10
  NWI = NWI * 5
  GO TO 20
10 ITW = 1
20 IF (NWI.NE.0) NWI = 2
C MAJOR LOOP FOR CALCULATING LAMINAR (K=1) AND TURBULENT (K=2) FLOW.
30 DO 400 K = 1,2
  IF (IPRINT.NE.1) GO TO 50

```

TEMP

TEMP 036
TEMP 037
TEMP 038
TEMP 039
TEMP 040
TEMP 041
TEMP 042
TEMP 043
TEMP 044
TEMP 045
TEMP 046
TEMP 047
TEMP 048
TEMP 049
TEMP 050
TEMP 051
TEMP 052
TEMP 053
TEMP 054
TEMP 055
TEMP 056
TEMP 057
TEMP 058
TEMP 059
TEMP 060
TEMP 061
TEMP 062
TEMP 063
TEMP 064
TEMP 065
TEMP 066
TEMP 067
TEMP 068
TEMP 069
TEMP 070

TEMP

TEMP 071
TEMP 072
TEMP 073
TEMP 074
TEMP 075
TEMP 076
TEMP 077
TEMP 078
TEMP 079
TEMP 080
TEMP 081
TEMP 082
TEMP 083
TEMP 084
TEMP 085
TEMP 086
TEMP 087
TEMP 088
TEMP 089
TEMP 090
TEMP 091
TEMP 092
TEMP 093
TEMP 094
TEMP 095
TEMP 096
TEMP 097
TEMP 098
TEMP 099
TEMP 100
TEMP 101
TEMP 102
TEMP 103
TEMP 104
TEMP 105

```

WRITE (TAPEOT,40)
NPRY = NPRT + 1
FORMAT(1M)
50 KT = 0
IFLOW = K
IMW = 0
HAW = H2 + RP(K)*(HTOT - H2)
TC1 = 100.
IF (NMI = 1)80,60,70
60 HW = HAW
IMW = 1
GO TO 210
70 TC1 = TR(K+4)
IF (TC1.GT.7000.) TC1 = 7000.
GO TO 210
80 ITW = 1
C NOTE, FOR REAL GAS EQUILIBRIUM TM IDEAL GAS DONE FIRST.
TR1 = TC1
QC1 = QC(TC1)
QR1 = RADK*1.E+8
QR2 = QC1
TR2 = (QR2/RADK)**0.25
TC2 = HAW/SCP
IF (TR2 .LT. TC2) GO TO 90
QC2 = 0.0
TR2 = TC2
QR2 = RADK*TR2**4
GO TO 100
90 TC2 = TR2
QC2 = QC(TC2)

C ITERATION CYCLE FOR TM
100 KT = KT + 1
C**CHECK IF TEMPERATURE ITERATIONS TO BE PRINTFD.
IF (IPRINT.NE.1) GO TO 120

```

TEMP

TEMP

```

WRITE (TAPEOT,110) KT,TC1,TR1,TC2,TR2,ITM,GC1,QR1,QC2,QR2
110 FORMAT(1M,2X0HKT =,I4,5X5HTC1 =,E13.6,5X5HTR1 =,E13.6,
1 5X5HTC2 =,E13.6,5X5HTR2 =,E13.6, /3X4HITM=,I4,
2 5X5HGC1 =,E13.6,5X5HQR1 =,E13.6,5X5HQC2 =,E13.6,
3 5X5HQR2 =,E13.6)
NPRT = NPET + 1

C CHECK IF ALLOWABLE NUMBER OF ITERATIONS EXCEEDED.
C :20 CONTINUE
IF (KT.GT.KTMAX) GO TO 200
DQC = QC1-QC2
DQR = QR2-QR1
DTC = TC1-TC2
DTR = TR2-TR1
C LINEAR SOLUTION (OR INTERCEPT FOR NEXT GUESS.
1 TC1 = ((QR1*TR2 - QR2*TR1)*DTC + (QC1*TC2 - QC2*TC1)*DTR)/
1 (DQC*DTR - DQR*DTC)

C CALCULATE HEATING RATES AND CHECK CONVERGENCE.
IF (TC1.LT.0.0) GO TO 140
130 TR1 = TC1
QR1 = RADK*TR1**4
QC1 = GC(TC1)
IF (ABS(1. - QC1/QR1).LE.EPST) GO TO 140
C NO SOLUTION, INITIATE NEXT CYCLE.
IF (QC1.GT.0.0) GO TO 170

C GC1 NEGATIVE, SPECIAL INITIALIZATION USED.
140 KSUB = 1
150 QC2 = 0.0
TC2 = HAW/GCP
TR2 = TC2
QR2 = RADK*TR2**4
IF (KSUB.NE.1) GO TO 100
TC1 = TC2

```

TEMP 106
TEMP 107
TEMP 108
TEMP 109
TEMP 110
TEMP 111
TEMP 112
TEMP 113
TEMP 114
TEMP 115
TEMP 116
TEMP 117
TEMP 118
TEMP 119
TEMP 120
TEMP 121
TEMP 122
TEMP 123
TEMP 124
TEMP 125
TEMP 126
TEMP 127
TEMP 128
TEMP 129
TEMP 130
TEMP 131
TEMP 132
TEMP 133
TEMP 134
TEMP 135
TEMP 136
TEMP 137
TEMP 138
TEMP 139
TEMP 140

TEMP

TEMP

141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175

```

160 TC1 = 0.5*TC1
    QC1 = QC(TC1)
    TR1 = TC1
    QR1 = RADK*TR1**4
    IF (QC1.GT.QR1) GO TO 100
    IF (KSUB.EQ.5) GO TO 100
    KSUB = KSUB + 1
    GO TO 160

C   QC1 POSITIVE, CONTINUE INITIALIZATION OF NEXT CYCLE.
170 QR2 = QC1
    TR2 = (QR2/RADK)**0.25
    TC2 = TR2
    QC2 = QC(TC2)
    IF (ABS(1. - QC2/QR2).LE.EPST) GO TO 180
    IF (QC2.GT.0.0) GO TO 100
    KSUB = 2
    GO TO 150

C   SOLUTION OBTAINED
180 TC1 = TC2

C   CHECK IF REAL GAS SOLUTION DESIRED.
190 IF ((ITW.EQ.2).OR.(NW.LT.3)) GO TO 210
    IF ((NW.EQ.6).OR.(NW.EQ.7)) GO TO 210

C   DETERMINE REAL GAS SOLUTION
    ITW = 2
    KT = 0
    GO TO 130

C   EXCEEDED ALLOWABLE ITERATIONS (KTMAX), AVERAGE LAST TWO VALUES
200 CONTINUE
    TC1 = (TC1 + TC2)*0.5
    GO TO 190
  
```

TEMP

TEMP

TEMP 176
 TEMP 177
 TEMP 178
 TEMP 179
 TEMP 180
 TEMP 181
 TEMP 182
 TEMP 183
 TEMP 184
 TEMP 185
 TEMP 186
 TEMP 187
 TEMP 188
 TEMP 189
 TEMP 190
 TEMP 191
 TEMP 192
 TEMP 193
 TEMP 194
 TEMP 195
 TEMP 196
 TEMP 197
 TEMP 198
 TEMP 199
 TEMP 200
 TEMP 201
 TEMP 202
 TEMP 203
 TEMP 204
 TEMP 205
 TEMP 206
 TEMP 207
 TEMP 208
 TEMP 209
 TEMP 210

TEMP

```

C   CALCULATE GC AT FINAL TM VALUE TO SET QUANTITIES IN COMMON,
210 GC1 = GC(TC1)
220 RT(K) = HAM/GCP
    IF(NM,EO,1,CR, NM,EO,4) TC1 = RT(K)
    TR(K+4) = TC1
    TR(K+6) = HW
    TR(K+8) = HAM
    TS(K) = TST1*FS(3)
C   RT AND TS ARE CORRECT ONLY FOR AN IDEAL GAS.
    GO TO (230,370),K
230 CFLLOC = CKU*ROMURA**2,0*PRAN**(2./3,)/(BS(1)*BS(7))
    RE(K) = (0.664*BS(3)/(TS(K)*CFLLOC))**2
    NW1 = 0
C
C   THE FOLLOWING CARDS ARE FOR LOCAL CF PRINTOUT ONLY,
C   LAMINAR FLOW
    CF1RE1 = 0.664*SQRT((BS(7)/FS(7))**3)*ROMURA
    CF1 = CF1RE1/SQRT(FS(8)*ELLOC)
    CF(1) = CF1
    ROPA = ROMURA**2
    HAM1 = HAM/H1
    NW1 = NW + 1
C
C   CHECK IF PRINTOUT OF LOCAL SKIN FRICTION CHARACTERISTICS DESIRED,
    IF ((IPRINT.LT,1).OR,(IPRINT.GT,2)) GO TO 380
C
240 NPRT = NPRT + 2
    GO TO (250,250,250,270,270,270,290,290,310,310),NW1
250 WRITE (TAPEOT,260) NW
260 FORMAT(1H0,2X3HNW=,I2,3X32HIDEAL GAS, REF, T/S=C SOLUTION,.)
    GO TO 330
270 WRITE (TAPEOT,280) NW
280 FORMAT(1H0,2X3HNW=,I2,3X32HREAL GAS, REF, H/S=C SOLUTION,.)
    GO TO 330

```

TEMP

```

290 WRITE (TAPEOT,300) NW
300 FORMAT(1M0,2X3HNWB,I2,3X35WIDEAL GAS, REF, T/REF, 7 SOLUTION,)
   GO TO 330
310 WRITE (TAPEOT,320) NW
320 FORMAT(1M0,2X3HNWB,I2,3X35WREAL GAS, REF, H/REF, 4 SOLUTION,)
330 WRITE (TAPEOT,340) KT,TC!,CF1,CFIRE1,PORA,TST1,MAWHI
   NPRT = NPRT + 1
340 FORMAT(1M , 2X3MKT, I2, 3X6HTWEG =,F7"1,1HR, 3X5HCF1 =,E13,6,
1 3X10MCF1(RE1) =,F9,6,3X9W ROMURA =,F9,5, 3X7HH*/HI =,F9,4,
2 3X8HHAW/HI =,F9,4)
   GO TO 380
C TURBULENT FLOW
350 IF (RET,LT,2540,0) GO TO 360
   CF1 = CFTLOC*RS(1)/FS(1)*(RS(7)/FS(7))**2
   CF(2) = CF1
   CFIRE1 = CF1*(FS(8)*C.LLOC)**0,2
360 RORA = 1,0/FC
   MAWHI = MAW/H:
   IF (IPRINT = 1)380,240,330
C THIS FNDS PRINTOUT CARDS.
370 RE(K) = RET
   IF (NW1,NE,0) GO TO 350
380 IF ((K.EQ,2).OR.(RE(1).GT,6570.)) GO TO 400
C TURBULENT FLOW SAME AS LAMINAR.
   RE(2) = RE(1)
   TS(2) = TS(1)
   RT(2) = RT(1)
   TR(6) = TR(5)
   TR(8) = TR(7)
   TR(10) = TR(9)
   CF(2) = CF(1)
C IF ((IPRINT,LT,1),OR,(IPRINT,GT,2)) GO TO 410

```

TEMP

TEMP

```
TEMP 246  
TEMP 247  
TEMP 248  
TEMP 249  
TEMP 250  
TEMP 251  
TEMP 252  
TEMP 253  
TEMP 254
```

```
WRITE (TAPEOT,390)  
390 FORMAT(1H0, 2X45HREYNOLDS NUMBEK BELOW CUTOFF, TURBULENT FLOW ,  
16H$AME AS LAMINAR.)  
1 NPRT = NPRT + 2  
C  
GO TO 410  
400 CONTINUE  
410 RETURN  
END
```

TEMP

ROMU

ROMU 001
 ROMU 002
 ROMU 003
 ROMU 004
 ROMU 005
 ROMU 006
 ROMU 007
 ROMU 008
 ROMU 009
 ROMU 010
 ROMU 011
 ROMU 012
 ROMU 013
 ROMU 014
 ROMU 015
 ROMU 016
 ROMU 017
 ROMU 018
 ROMU 019
 ROMU 020
 ROMU 021
 ROMU 022
 ROMU 023
 ROMU 024
 ROMU 025
 ROMU 026
 ROMU 027
 ROMU 028
 ROMU 029
 ROMU 030
 ROMU 031
 ROMU 032
 ROMU 033
 ROMU 034
 ROMU 035

ROMU

```

FUNCTION ROMU(HS,P2)
C
C THIS FUNCTION HAS THREE ENTRIES. THE FIRST, ROMU(HS,P2), CALCULATES
C THE DENSITY-VISCOSITY PRODUCT FOR EQUILIBRIUM AIR AT THE INPUT
C ENTHALPY (HS) AND PRESSURE (P2).
C RANGE, HS TO ABOUT 4.E+6 (FT/SEC)**2 AND P2 .GT. 1.F-4
C AND .LT. 10. EXTRAPOLATION FOR P2 OUTSIDE THIS.
C THE SECOND, ENTHAL(TW,PW), CALCULATES THE ENTHALPY CORRESPONDING
C TO THE INPUT TEMPERATURE (TW) AND PRESSURE (PW).
C RANGE, TW TO ABOUT 8000. OR (PROGRAM HAS CUTOFF AT
C 7000. OR) AND PW SAME AS P2 ABOVE.
C THE THIRD, ROM(HS,P2), CALCULATES THE DENSITY (SLUGS/FT**3)
C FOR EQUILIBRIUM AIR AT THE INPUT ENTHALPY (HS) AND PRESSURE (P2).
C RANGE, SAME AS FOR ROMU ABOVE.
C ALL ENTRIES REQUIRE BLOCK DATA FOR COMMON/PROP/ AND FUNCTION
C POLY. THE PROPERTIES BASED ON AEDC TR-65-SR AND HANSENS VISCOSITY.
C DETAILS OF THE PROPERTY FITS ARE GIVEN IN DOUGLAS REPORT LB-32706.
C*****D. N. SMYTH PROGRAM AUTHOR*****
C
C COMMON /PROP/ FH(135), FR(135)
C DIMENSION I4(3)
C DATA PRF/2117.36/
C HI = HS*.0E+9
C
C DETERMINE ENTHALPY RANGE AT 10.0ATM
C IRMI = 1
C IF (HI.LT.0.042) GO TO 10
C IRMI = 10
C IF (HI.LT.0.1768) GO TO 10
C IRMI = 46
C IF (HI.LT.0.650) GO TO 20
C IRMI = 55
  
```

ROMU

ROMU 036
 ROMU 037
 ROMU 038
 ROMU 039
 ROMU 040
 ROMU 041
 ROMU 042
 ROMU 043
 ROMU 044
 ROMU 045
 ROMU 046
 ROMU 047
 ROMU 048
 ROMU 049
 ROMU 050
 ROMU 051
 ROMU 052
 ROMU 053
 ROMU 054
 ROMU 055
 ROMU 056
 ROMU 057I
 ROMU 058C
 ROMU 059
 ROMU 060
 ROMU 061
 ROMU 062
 ROMU 063
 ROMU 064
 ROMU 065
 ROMU 066
 ROMU 067
 ROMU 068
 ROMU 069
 ROMU 070

```

    IF (H1.GT.1.10) IRM1 = 64
  C  DETERMINE ENTHALPY RANGE AT 10.0**^1 ATM
  10 IF (H1.GT.0.1768) GO TO 20
    IRM2 = IRM1
    H1 = H1*10.0
    GO TO 30
  20 IRM2 = 19
    IF (H1.LT.0.580) GO TO 30
    IRM2 = 28
    IF (H1.GT.0.980) IRM2 = 37
  30 N = 6
    PBAR = P2/PREF
    PBARLG = ALOG10(PBAR)
    ROMJ = -0.2*((PRARLG-1.0)*POLY(FH,IRM2,H1,N) - (PBARLG+4,0))*
      1 POLY(FR,IRM1,H1,N)
    ROMJ = ROMU*PBAR*1.0E-9
    RETURN
  C
  C  DETERMINE ENTHALPY FOR GIVEN TEMPERATURE (MAXIMUM 7000.0R)
  C  ENTRY ENTHAL(TH,PW)
  C  ENTRY ENTHAL
    TW = HS
    PW = P2
    N = 8
    IF (TW.GT.7000.0) TW = 7000.0
    TWX = TW*1.0E+4
    J = 1
    IF (TW.GT.2700.) GO TO 40
    IM1 = 1
    ROMU = POLY(FH,IM1,TWX,N)*1.0E+8
    RETURN
  C  40 PBAR = PW/PREF
  
```

ROMU

ROMU

ROMU	071
ROMU	072
ROMU	073
ROMU	074
ROMU	075
ROMU	076
ROMU	077
ROMU	078
ROMU	079
ROMU	080
ROMU	081
ROMU	082
ROMU	083
ROMU	084
ROMU	085
ROMU	086
ROMU	087
ROMU	088
ROMU	089
ROMU	090
ROMU	091
ROMU	092
ROMU	093
ROMU	094
ROMU	095
ROMU	096
ROMU	097
ROMU	098
ROMU	099
ROMU	100
ROMU	101
ROMU	102
ROMU	103
ROMU	104
ROMU	105

ROMU

```

C      PBARLG = ALOG10(PBAR)
C      DETERMINE THREE PBARLG VALVES TO USE IN QUADRATIC INTERPOLATION
      IF (PBARLG.GT.-3.0) GO TO 50
      PBAR1 = -4.0
      PBAR2 = -3.0
      PBAR3 = -2.0
      GO TO 80
50 IF (PBARLG.GT.-2.0) GO TO 60
      PBAR1 = -3.0
      PBAR2 = -2.0
      PBAR3 = -1.0
      GO TO 100
60 IF (PBARLG.GT.-1.0) GO TO 70
      PBAR1 = -2.0
      PBAR2 = -1.0
      PBAR3 = 0.0
      GO TO 120
70 PBAR1 = -1.0
      PBAR2 = 0.0
      PBAR3 = 1.0
      GO TO 140

C      DETERMINE TEMP. RANGE AT PBARLG = -4
80 IH(J) = 10
      IF (TW.LT.,4200.) GO TO 90
      IH(J) = 19
      IF (TW.GT.,5500.) IH(J) = 28
90 J = J + 1

C      DETERMINE TEMP. RANGE AT PBARLG = -3
100 IH(J) = 37
      IF (TW.LT.,4540.) GO TO 110
      IH(J) = 46
      IF (TW.GT.,6120.) IH(J) = 55

```

ROMU

```

110 J = J + 1
C DETERMINE TEMP. RANGE AT PBARLG = -2
120 IH(J) = 64
   IF (TM.LT.5220.) GO TO 130
   IH(J) = 73
   IF (TM.GT.6660.) IH(J) = 82
130 J = J + 1
   IF (J.GT.3) GO TO 150
C DETERMINE TEMP. RANGE AT PBARLG = -1
140 IH(J) = 91
   IF (TM.GT.5580.) IH(J) = 100
   J = J + 1
   IF (J.GT.3) GO TO 150
C DETERMINE TEMP. RANGE AT PBARLG = 0
150 IH(J) = 109
   IF (TM.GT.6300.) IH(J) = 118
   J = J + 1
   IF (J.GT.3) GO TO 150
C DETERMINE TEMP. RANGE AT PBARLG = 1
   IH(J) = 127
C CALCULATE ENTHALPY
150 IH1 = IH(1)
   IH2 = IH(2)
   IH3 = IH(3)
   HC1 = (PBARLG - PBAR2)*(PBARLG - PBAR3)*0.5
   HC2 = (PBARLG - PBAR1)*(PBARLG - PBAR3)
   HC3 = (PBARLG - PBAR1)*(PBARLG - PBAR2)*0.5
   ROMU = (HC1*POLY(FH,IH1,TWX,N) - HC2*POLY(FH,IH2,TWX,N)
1       + HC3*POLY(FH,IH3,TWX,N))*1.0E+8
   RETURN
C

```

ROMU 106
ROMU 107
ROMU 108
ROMU 109
ROMU 110
ROMU 111
ROMU 112
ROMU 113
ROMU 114
ROMU 115
ROMU 116
ROMU 117
ROMU 118
ROMU 119
ROMU 120
ROMU 121
ROMU 122
ROMU 123
ROMU 124
ROMU 125
ROMU 126
ROMU 127
ROMU 128
ROMU 129
ROMU 130
ROMU 131
ROMU 132
ROMU 133
ROMU 134
ROMU 135
ROMU 136
ROMU 137
ROMU 138
ROMU 139
ROMU 140

ROMU

ROMU

ROMU 141
ROMU 142
ROMU 143C
ROMU 144I
ROMU 145
ROMU 146
ROMU 147
ROMU 148
ROMU 149
ROMU 150
ROMU 151
ROMU 152
ROMU 153
ROMU 154
ROMU 155
ROMU 156
ROMU 157
ROMU 158
ROMU 159
ROMU 160
ROMU 161
ROMU 162
ROMU 163
ROMU 164
ROMU 165
ROMU 166
ROMU 167
ROMU 168
ROMU 169
ROMU 170

```
C C DETERMINE DENSITY FOR GIVEN ENTHALPY AND PRESSURE.
C   ENTRY ROM
C   ENTRY ROM(HS,P2)
C   H1 = HS*1.0E-8
C   N = 5
C   DETERMINE ENTHALPY RANGE AT 10.0 ATM.
C   IRN1 = 73
C   IF (H1.GT.0.1768) GO TO 160
C   ROMU = 0.12336898/POLY(FR,IRN1,H1,N)
C   GO TO 190
160 N = 8
C   IRN1 = 109
C   IF (H1.LT.0.503) GO TO 170
C   IRN1 = 118
C   IF (H1.GT.0.983) IRN1 = 127
C   DETERMINE ENTHALPY RANGE AT 10.0**=4 ATM.
170 IRN2 = 82
C   IF (H1.LT.0.542) GO TO 180
C   IRN2 = 91
C   IF (H1.GT.0.889) IRN2 = 100
180 PBAR = P2/PRFF
C   PBARLG = ALOG10(PBAR)
C   ROMU = -0.2*((PBARLG - 1.0)*POLY(FR,IRN2,H1,N) -
C   (PBARLG + 4.0)*POLY(FR,IRN1,H1,N))
190 ROMU = ROMU*PBAR*1.0E-3
C   RETURN
C   END
```

ROMU

GC

```

FUNCTION QC(TW)
GC
GC
C CALCULATES THE AERODYNAMIC HEATING AT THE GIVEN
C WALL TEMP.(TW) IN LAMINAR (IFLOW = 1) AND TURBULENT
C (IFLOW = 2) FLOW OF EITHER AN IDEAL GAS (ITW = 1)
C OR A REAL GAS (ITW = 2). REFERENCE TEMPERATURE OR
C REFERENCE ENTHALPY USED FOR LAMINAR FLOW. SPALDING=CHI (ITURB = 1)
C OR REFERENCE TEMPERATURE/REFERENCE ENTHALPY (ITURB = 2) USED
C FOR TURBULENT FLOW.
C*****D. N. SMYTH PROGRAM AUTHOR*****
C
C
C
COMMON /FSBS/FS,BS
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /TEMPOC/HAW,M2,M1,MW,CKU,FC,FRX,RFT,ELLOC,GCP, YS71,ROMURA
COMMON /FLAG2/ITW,IHW,IFLOW,ITURB,CFTLNC
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
DIMENSION TITL(15),BS(8),FS(8)
INTEGER CASE,PAGE,ERROR
C
C
C
C MONAGHAN REFERENCE CONDITION COEFFICIENTS
A2 = 0.468*PRAN**0.3333
A1 = 1.0 - A2
A2 = A2 * 0.273*SQRT(PRAN)
PW = PS(2)
C
C CHECK FOR LAMINAR OR TURBULENT.
IF (IFLOW=2)20,10,20
10 GO TO (130,170), ITURB
C LAMINAR FLOW. CHECK IF ENTHALPY INPUT.
20 IF (IHW)40,40,30
30 GO TO (60,110), ITW
40 IF (ITW,EG.2) GO TO 100

```

GC

QC

QC	036
QC	037
QC	038
QC	039
QC	040
QC	041
QC	042
QC	043
QC	044
QC	045
QC	046
QC	047
QC	048
QC	049
QC	050
QC	051
QC	052
QC	053
QC	054
QC	055
QC	056
QC	057
QC	058
QC	059
QC	060
QC	061
QC	062
QC	063
QC	064
QC	065
QC	066
QC	067
QC	068
QC	069
QC	070

```

C
C REFERENCE TEMP. SOLUTION
50 HW = GCP*TW
60 TST1 = (A1*HW + A2*HAM + (1.-A1)*A2)*M2)/H1
   TCT1 = 198.6/FS(3)
   IF (FS(3).GT.AV1) GO TO 90
   IF (TST1*FS(3).GT.AV1) GO TO 70
   VISRA = TST1*AV3
   GO TO 80
70 VISRA = 2.270E+8*SQRT(TST1*FS(3))/(1.+ TCT1/TST1)/ FS(5)
80 ROMURA = SQRT(8S(2)/FS(2)*VISRA/TST1)
   GO TO 120
90 ROMURA = SQRT(SQRT(TST1)*(1.+TCT1)/(TST1+TCT1)*8S(2)/FS(2))
   GO TO 120

C
C REFERENCE ENTHALPY SOLUTION.
100 HW = ENTHAL(TW,PW)
110 MSTAR = A1*HW + A2*HAM + (1.-A1)*A2)*M2
   ROMURA = SQRT(ROMU(MSTAR,PW)/(FS(1)*FS(5)))
   TST1 = MSTAR/H1
120 QC = CKU*ROMURA*(HAM = HW)
   RETURN

C
C TURBULENT FLOW, SPALDING-CHI METHOD
130 IF (IHW.GT.0) GO TO 160
   TW1 = TW
   IF(TW1.LT.100.0) TW1= 100.0
   GO TO (140,150),ITW

C IDEAL GAS SOLUTION
140 HW = GCP*TW1
   GO TO 160

C REAL GAS SOLUTION
150 HW = ENTHAL(TW1,PW)
160 A = HAM/M2 = 1.

```

QC

OC

OC 106

END

OC

POLY

POLY 001
POLY 002
POLY 003
POLY 004
POLY 005
POLY 006
POLY 007
POLY 008
POLY 009
POLY 010
POLY 011
POLY 012

FUNCTION POLY(A,I,HX,N)
DIMENSION A(135)

C THIS FUNCTION GENERATES AN N-TH ORDER POLYNOMIAL
C IN HX WITH COEFFICIENTS A(K) STARTING WITH K=I.
C

POLY = A(I)
DO 10 J = 1,N
K = I + J
10 POLY = POLY + A(K)*HX**J
RETURN
END

POLY

BLKDAT

BLKD 001
 BLKD 002
 BLKD 003
 BLKD 004
 BLKD 005
 BLKD 006
 BLKD 007
 BLKD 008
 BLKD 009
 BLKD 010
 BLKD 011
 BLKD 012
 BLKD 013
 BLKD 014
 BLKD 015
 BLKD 016
 BLKD 017
 BLKD 018
 BLKD 019
 BLKD 020
 BLKD 021
 BLKD 022
 BLKD 023
 BLKD 024
 BLKD 025
 BLKD 026
 BLKD 027
 BLKD 028
 BLKD 029
 BLKD 030
 BLKD 031
 BLKD 032
 BLKD 033
 BLKD 034
 BLKD 035

BLOCK DATA

C C THIS SUBROUTINE INITIALIZES INTO COMMON/PROP/ THE COEFFICIENT
 C ARRAYS REQUIRED BY FUNCTION ROMU (AR03) TO DETERMINE THE
 C REAL EQUILIBRIUM AIR PROPERTIES. OTHER PROGRAM PARAMETERS ARE ALSO
 C INITIALIZED.

C C COMMON /PROP/ FM,FR
 C DIMENSION FH(135),FR(135),FH1(108),FHP(27),FR1(72),FR2(63)
 C EQUIVALENCE (FH(1),FH1(1)),(FH(109),FH2(1)),
 C (FR(1),FR1(1)), (FR(73),FR2(1))

C C DATA
 C FM1 / 0.0, .60181771, -.22228717, 2.5953429,
 1 -2.7853922, .9, 7.22052, 23.289510, -.12, 1.99244, -.2, 5.817427,
 2 -.1293540, -.2, 9.966888, 48.010840, -.180, 7.7242, 249.48974,
 3 -51.244932, 0.0, 0.0, 0.0, -.6, 5.939838, 19.010757, 19.992531,
 4 -39.975681, -.112, 8.9466, 156.16378, 0.0, 0.0, 0.0, 2.2022949,
 5 -14.101200, 23.004509, 52.890407, -.165, 0.8756, 113.04871, 0.0,
 6 0.0, 0.0, .20293060, .71240021, 11.699213, -47.629621,
 7 42.913642, 30.631199, 0.0, 0.0, 0.0, 1.8692814, -22.097879,
 8 56.153394, -16.042275, 36.738951, -.263, 4.9237, 239.87072, 0.0,
 9 0.0, -.15686170, 1.1830883, -2.0713376, 19.029276, -26.810587,
 X -9.9053574, 24.334188, 0.0, 0.0, .92262030, -.8, 5.032999,
 A 20.492311, 45.587738, -119.90206, -509.58200, 1721.3813,
 B -1337.6228, 0.0, -2.6782862, 1.2865940, 22.104893, -12.395676,
 C -26.121859, -12.897718, 38.794573, 0.0, 0.0, -1.6135321,
 D 5.3650713, 2.4832548, -9.7412082, 1.2468173, .39646271, 4.5573527
 E, 0.0, 0.0, .44553169, -3.0252481, 2.6196596, 50.144314,
 F -151.34585, 133.67463, -154.12089, 535.69899, -494.43566,
 G -3.0416265, 12.560193, -20.33612, 17.381598, 23.843668,
 H -64.498613, 36.134646, 0.0, 0.0

BLKDAT

BLKDAT

BLKD 036
 BLKD 037
 BLKD 038
 BLKD 039
 BLKD 040
 BLKD 041
 BLKD 042
 BLKD 043
 BLKD 044
 BLKD 045
 BLKD 046
 BLKD 047
 BLKD 048
 BLKD 049
 BLKD 050
 BLKD 051
 BLKD 052
 BLKD 053
 BLKD 054
 BLKD 055
 BLKD 056
 BLKD 057
 BLKD 058
 BLKD 059
 BLKD 060
 BLKD 061
 BLKD 062
 BLKD 063
 BLKD 064
 BLKD 065
 BLKD 066
 BLKD 067
 BLKD 068
 BLKD 069
 BLKD 070

C DATA
 1 -20.017959, 155.77436, -439.43620, 709.71808, -671.23004,
 2 317.22690, 1.5380637, -11.161238, 32.383428, 2.5847156,
 3 -196.67668, 370.82431, -218.12898, 0.0, 0.0, 1.3507331,
 4 -5.0193304, 6.0404391, 4.1730680, -13.136917, 6.7913235, 0.0,
 5 0.0, 0.0, .42795709, -.56949512, .48404761, -.11638753,
 6 -.74634946E=2, .12505798E=1, -.27304295E=2, .19434979E=3, 0.0,
 71.3644197, -9.5027751, 34.791129, -.68.553220, 69.547355,
 8 -28.523442, 0.0, 0.0, .48102262, -.91625506, .62421014,
 9 .57730033, -.1.0682441, .42044702, 0.0, 0.0, 0.0, .42220834,
 X -.61202419, .44184292, -.15420936, .18083084E=1, .36418043E=2,
 A -.12611833E=2, .10222056E=3, 0.0 /
 FM2 / 1.199213, -.65116600, 3.8928011,
 1 -.45505770, 5.3407127, -21.592914, 33.774256, 58.270657,
 2 .34603226, -.79201527, -.79826544, 5.2064374,
 3 -1.2818503, 4.3016173, -.6.9417639, 0.0, 0.0, .11764422,
 4 -.62580124, 3.5461409, 4.8925829, -39.857784, 71.680475,
 5 -48.794024, 9.9564413, 0.0 /,
 6 FR1 / 1.0490322, -.31228101, .90103454E=1, .6.7117799,
 7 18.184363, -13.551560, 0.0, 0.0, 1.1329366, -.94702472,
 8 .55849455, -.92146834E=1, -.50705935E=1, .17519852E=1, 0.0, 0.0, 0.0,
 9 .88808527, -3.9763502, 14.251132, -27.521840, 26.791510,
 X -10.436079, 0.0, 0.0, -.14564908, 3.0171600, .5.9999473,
 A 4.6748398, -1.2742339, 0.0, 0.0, 0.0, .37102156, -.13586302,
 B .44241289E=1, -.78199161E=2, .52479786E=3, 0.0, 0.0, 0.0, 0.0,
 C .77497574, -2.1673001, 4.0652322, -2.7726303, -.1.814257,
 D 1.6918670, 0.0, 0.0, .52865913, -.58894771, .58988212,
 E -.02423488, .19163722, -.38356068E=1, 0.0, 0.0, 0.0, .48598990,
 F -.41125640, .25298705, -.82910932E=1, .13654598E=1, .89113877E=3,
 G 0.0, 0.0, 0.0 /

C

C C

C DATA
 1 -20.017959, 155.77436, -439.43620, 709.71808, -671.23004,
 2 317.22690, 1.5380637, -11.161238, 32.383428, 2.5847156,
 3 -196.67668, 370.82431, -218.12898, 0.0, 0.0, 1.3507331,
 4 -5.0193304, 6.0404391, 4.1730680, -13.136917, 6.7913235, 0.0,
 5 0.0, 0.0, .42795709, -.56949512, .48404761, -.11638753,
 6 -.74634946E=2, .12505798E=1, -.27304295E=2, .19434979E=3, 0.0,
 71.3644197, -9.5027751, 34.791129, -.68.553220, 69.547355,
 8 -28.523442, 0.0, 0.0, .48102262, -.91625506, .62421014,
 9 .57730033, -.1.0682441, .42044702, 0.0, 0.0, 0.0, .42220834,
 X -.61202419, .44184292, -.15420936, .18083084E=1, .36418043E=2,
 A -.12611833E=2, .10222056E=3, 0.0 /
 FM2 / 0.0, 1.6627798, 1.1792385,
 1 -20.017959, 155.77436, -439.43620, 709.71808, -671.23004,
 2 317.22690, 1.5380637, -11.161238, 32.383428, 2.5847156,
 3 -196.67668, 370.82431, -218.12898, 0.0, 0.0, 1.3507331,
 4 -5.0193304, 6.0404391, 4.1730680, -13.136917, 6.7913235, 0.0,
 5 0.0, 0.0, .42795709, -.56949512, .48404761, -.11638753,
 6 -.74634946E=2, .12505798E=1, -.27304295E=2, .19434979E=3, 0.0,
 71.3644197, -9.5027751, 34.791129, -.68.553220, 69.547355,
 8 -28.523442, 0.0, 0.0, .48102262, -.91625506, .62421014,
 9 .57730033, -.1.0682441, .42044702, 0.0, 0.0, 0.0, .42220834,
 X -.61202419, .44184292, -.15420936, .18083084E=1, .36418043E=2,
 A -.12611833E=2, .10222056E=3, 0.0 /

C C

BLKDAT

BLKDAT

BLKD 071

BLKDAT

END

CFINPT

```

C
C
SURROUTINE CFINPT(INP, LINTAB)

COMMON /GDATA/LINT,J,SYMFCT,IORN,IGTYPE,L
COMMON /ABDATA/NAB,ALPHA(20),BETA(20),ROL(20),CDELTA(20),OI(20),
1 RI(20),PI(20)
COMMON /EXEC/CASE,TITLE,PAGE,ERROR
COMMON /TAPE/TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPFD,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON/C1 /INRM,ISURF,IX,IY,IZ,U1,UL,V1,VL,CR,CT,CHX,NP,N3,
1 XJ,YJ,ZJ,AP11,AP12,AP13,AP21,AP22,AP23,AP31,AP32,AP33,
2 FLOWC(2),XI(500),YI(500),BI(503,2),XKF(503),AA(2477),
A XB(4), YB(4), ZB(4)
3 NDSSET,IABSET,IR,LORG(20),ISR(20),NS,TSFR,IVIS,IFTYP,IFLOW,
4 BX(2,20),BY(2,20),NB,CUM(4)

C
DIMENSION YTITLE(15), IMP(20), LCOND(3), DINFL(6)
DIMENSION YTITLEM(10), INTAB(9), LOAB(20), TITLES(10), TITLEA(10),
1 YTLER(10),E4(12),E2(13),DAT(6),
3 E3(17), ISF(5), FDATA(34), XP(4,6)

C
INTEGER FRROR,PAGE,CARF,SYMFCT
INTEGER TAPEIN,TAPEO,TAPEA,TAPEB,TAPEC,TAPFD,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK

C
EQUIVALENCE (FDATA(1), LCOND(1)), (FDATA(4), ELEM(1)),
1 (FDATA(29), DINFL(1))
DATA RC/0.1745329E-1/

C
WRITE(TAPEO,5)
5 FORMAT(1H1,45HPREVIOUSLY SAVED SURFACE DATA WILL BE USED TO,
1 17H CALCULATE FORCES)
C FOR CHECKOUT SET IPRINT = 1
IPRINT = 0

```

CFINPT

```

CFIN 001
CFIN 002
CFIN 003
CFIN 004
CFIN 005
CFIN 006
CFIN 007
CFIN 008
CFIN 009
CFIN 010
CFIN 011
CFIN 012
CFIN 013
CFIN 014
CFIN 015
CFIN 016
CFIN 017
CFIN 018
CFIN 019
CFIN 020
CFIN 021
CFIN 022
CFIN 023
CFIN 024
CFIN 025
CFIN 026
CFIN 027
CFIN 028
CFIN 029
CFIN 030
CFIN 031
CFIN 032
CFIN 033
CFIN 034
CFIN 035

```

CFINPT

```

C C BEGIN ALPHA-BETA CYCLE
C C REWIND TAPEF
C C DO 2000 IAB = 1, NAB
C C ICT = 0
C C ISYM = 1
C C IF (IMP(IAB) .EQ. 1) GO TO 2000
C C INPUT REGION ID INFO AND NORMALIZATION FLAGS
C C 10 READ (TAPEIN, 20) LASTR, NDSET, IABSET, IR, INORM, ISURF, IPF,
C C 1 (ISR(I), I=1, 10), ISF
C C 20 FORMAT (I1, 3I2, 3I1, 10I2, 5I1)
C C WRITE (TAPEOT, 32) LASTR, NDSET, IABSET, IR, INORM, ISURF,
C C 1 IPF, (ISR(I), I=1, 10), ISF
C C 32 FORMAT (1H0, 5X, 6HLASTR, 12, 4X, 6HNDSET, 12, 4X, 7H IABSET, 12, 4X,
C C 1 4HIR, 2, 12, 4X, 6HINORM, 12, 4X, 6HISURF, 12, 4X,
C C 2 /1H, 4HIPF, 12, 4X4HISR, 10I3, 4X4HISF, 5I2/)
C C READ BOUNDARY POINTS FOR NORMALIZATION
C C DO 38 I=1, 4
C C 38 READ (TAPEIN, 50) XB(I), YB(I), ZB(I)
C C 50 FORMAT (3F10, 0)
C C READ IN MASTER DIRECTORY
C C IYTAG10 = 1
C C CALL READMS (10, TITLEM, 10, ITAG10)
C C READ(10#ITAG10) TITLEM
C C IYTAG10 = 2
C C CALL READMS (10, INTAB, 9, ITAG10)
C C READ(10#ITAG10) INTAB
C C CHECK DATA SET NUMBER
C C IF (INTAB(1) .GE. NDSET) GO TO 40

```

CFINPT

```

CFIN 036
CFIN 037
CFIN 038
CFIN 039
CFIN 040
CFIN 041
CFIN 042
CFIN 043
CFIN 044
CFIN 045
CFIN 046
CFIN 047
CFIN 048
CFIN 049
CFIN 050
CFIN 051
CFIN 052
CFIN 053
CFIN 054
CFIN 055
CFIN 056
CFIN 057
CFIN 058
CFIN 059
CFIN 060
CFIN 061
CFIN 062
CFIN 063C
CFIN 064I
CFIN 065
CFIN 066C
CFIN 067I
CFIN 068
CFIN 069
CFIN 076

```

CFINPY

```

39 WRITE (TAPEUT,39) N0SET,INTAR(1)
   FORMAT (10,9H#N0SET #,12,31H IS GREATER THAN THE NUMBER OF ,
1  Q4HDATA SETS ACTUALLY ON UNIT 10. PROGRAM HALT, )
   STOP

C   READ IN FLOW DATA SET DIRECTORY
40 ITAG10 = INTAB(N0SET + 4)
   CALL READMS (10,F4,12,ITAG10)
   READ (10X:ITAG10) E4
   DO 41 I=1,10
41 TITLFS(I) = F4(I)
   MACH = E4(11)
   NARS = E4(12)
   ITAG10 = ITAG10 + 1
   CALL READMS (10,LOAD,20,ITAG10)
   READ(10X:ITAG10) LOAD

C   READ IN FLOW REGION DIRECTORY FOR REQUESTED ALPHA=BETA SET
ITAG10 = LOAB(IARSET)
CALL READMS (10,E2,13,ITAG10)
READ (10X:ITAG10) E2
DO 42 I=1,10
42 TITLFA(I) = E2(I)
   ALPHAS = E2(11)
   RETAS = E2(12)
   NREG = E2(13)
   ITAG10 = ITAG10 + 1
   CALL READMS (10,LORG,20,ITAG10)
   READ(10X:ITAG10) LORG

C   READ IN REQUESTED FLOW REGIONS

```

CFIN 073
CFIN 072
CFIN 073
CFIN 074
CFIN 075
CFIN 076
CFIN 077
CFIN 078
CFIN 079C
CFIN 080I
CFIN 081
CFIN 082
CFIN 083
CFIN 084
CFIN 085
CFIN 086C
CFIN 087I
CFIN 088
CFIN 089
CFIN 090
CFIN 091C
CFIN 092I
CFIN 093
CFIN 094
CFIN 095
CFIN 096
CFIN 097
CFIN 098
CFIN 099C
CFIN 100I
CFIN 101
CFIN 102
CFIN 103
CFIN 104
CFIN 105

CFINPY

CFINPT

CFIN 106
 CFIN 107
 CFIN 108
 CFIN 109
 CFIN 110C
 CFIN 111I
 CFIN 112
 CFIN 113
 CFIN 114
 CFIN 115
 CFIN 116
 CFIN 117
 CFIN 118
 CFIN 119
 CFIN 120
 CFIN 121
 CFIN 122
 CFIN 123
 CFIN 124
 CFIN 125
 CFIN 126
 CFIN 127
 CFIN 128
 CFIN 129
 CFIN 130
 CFIN 131
 CFIN 132
 CFIN 133
 CFIN 134
 CFIN 135
 CFIN 136
 CFIN 137
 CFIN 138
 CFIN 139
 CFIN 140

CFINPT

```

C      ITAG10 = LORG(IR)
C      CALL READMS (10,E3,17,ITAG10)
C      READ (10,ITAG10) E3
C      DO 91 I=1,10
C      91 TITLER(I) = E3(I)
C      DO 92 I=1,5
C      92 CONTINUE
C      IDTYP(1) = E3(I+10)
C      IF TYP = IDTYP(1)
C      IF LWC = IDTYP(2)
C      IVIS = 0
C
C      THIS ROUTINE HAS BEEN TAKEN FROM A GENERAL PROGRAM
C      AND SPECIALIZED FOR SKIN FRICTION INTERPOLATION.
C      A CHECK IS MADE TO DETERMINE IF CF HAS BEEN
C      STORED. IF NOT, RUN IS HALTED.
C
C      IF (IFTYP .EQ. 4) GO TO 93
C      WRITE(TAPEOT, 90)
C      90 FORMAT(1M1,45H**** SKIN FRICTION DATA HAS NOT BEEN STORED ,
C      1 25HWITH THE STREAMLINE DATA./1M0,5X25H A REQUEST TO INTERPOLATE.
C      2 45H FOR CF AT ELEMENT CENTROIDS CANNOT BE MADE. /1M0, 5X,
C      3 15H RUN STOPPED. )
C      STOP
C
C      93 CONTINUE
C      IVIS = 1
C      IPF = 1
C      IF (ISF(1) .GT. 0) GO TO 94
C      ISF(1) = 1
C      ISF(2) = 2
C      ISF(3) = 3

```

CFINPT

CFIN 141
CFIN 142
CFIN 143
CFIN 144
CFIN 145C
CFIN 146I
CFIN 147
CFIN 148
CFIN 149
CFIN 150
CFIN 151
CFIN 152
CFIN 153
CFIN 154
CFIN 155
CFIN 156
CFIN 157
CFIN 158
CFIN 159
CFIN 160
CFIN 161
CFIN 162
CFIN 163
CFIN 164
CFIN 165
CFIN 166
CFIN 167
CFIN 168
CFIN 169
CFIN 170
CFIN 171
CFIN 172
CFIN 173
CFIN 174
CFIN 175

```
ISF(4) = 4
ISF(5) = 0
90 NSREG = E3(16)
ITAG10 = ITAG10 + 1
CALL READMS (10, DAT, 6, ITAG10)
READ (10, ITAG10) DAT
XO = DAT( 1)
YO = DAT( 2)
ZO = DAT( 3)
PSIO = DAT( 4)
THETO = DAT( 5)
PHIC = DAT( 6)

SINT = SIN(THETO*RC)
COST = COS(THETO*RC)
COSFS = COS(PSIO*RC)
SINPS = SIN(PSIO*RC)
SINP = SIN(PHIO*RC)
COSP = COS(PHIO*RC)
AP11 = COST*COSPS
AP12 = COST*SINPS
AP13 = SINT
AP21 = COSP*SINPS + SINP*SINT*COSPS
AP22 = COSP*COSPS + SINP*SINY*SINPS
AP23 = SINP*COST
AP31 = SINP*SINPS + COSP*SINT*COSPS
AP32 = SINT*COSPS + COSP*SINT*SINPS
AP33 = COSP*COST

C SET INDICES FOR NORMALIZING DATA
IF(INORM .LE. 0) GO TO 100
GO TO (110,150,140,150), INORM

C NORMALIZE M,R,T: A,R
100 IX = 4
```

CFINPT

CFINPT

CFIN 176
 CFIN 177
 CFIN 178
 CFIN 179
 CFIN 180
 CFIN 181
 CFIN 182
 CFIN 183
 CFIN 184
 CFIN 185
 CFIN 186
 CFIN 187
 CFIN 188
 CFIN 189
 CFIN 190
 CFIN 191
 CFIN 192
 CFIN 193
 CFIN 194
 CFIN 195
 CFIN 196
 CFIN 197
 CFIN 198
 CFIN 199
 CFIN 200
 CFIN 201
 CFIN 202
 CFIN 203
 CFIN 204
 CFIN 205
 CFIN 206
 CFIN 207
 CFIN 208
 CFIN 209
 CFIN 210

IY = 5
 IZ = 6
 GO TO 190

C NORMALIZE W.R.T. X,Y
 110 IX = 1
 IY = 2
 IZ = 3
 GO TO 190

C NORMALIZE W.R.T. X,Z
 130 IX = 1
 IY = 3
 IZ = 2
 GO TO 190

C NORMALIZE W.R.T. Y,Z
 140 IX = 2
 IY = 3
 IZ = 1
 GO TO 190

C NORMALIZE W.R.T. A,PHI
 150 IX = 4
 IY = 6
 IZ = 5
 GO TO 190

C CALCULATE NORMALIZING LENGTHS
 190 CONTINUE
 DO 95 I = 1,4
 XX = XB(I) * XO
 YY = YB(I) * YO
 ZZ = ZB(I) * ZO
 XP(I,1) = XX*AP11 + YY*AP12 + ZZ*AP13

CFINPT

CFINPT

211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245

```
XP(I,2) = XX*AP21 + YY*AP22 + ZZ*AP23
XP(I,3) = XX*AP31 + YY*AP32 + ZZ*AP33
XP(I,4) = XP(I,1)
XP(I,5) = SQRT(XP(I,2)**2 + XP(I,3)**2)
IF (XP(I,2) .EQ. 0.0) GO TO 194
XP(I,6) = ATAN2(XP(I,2), XP(I,3))
GO TO 95
194 XP(I,6) = 0.0
IF (XP(I,3) .GT. 0.0) XP(I,6) = 3.141592654
95 CONTINUE
C
C
U1 = XP(1,IX)
UL = XP(2,IX) = U1
CR = UL
IF (ISURF .FG. 1) GO TO 96
V1 = XP(3,IY)
VL = XP(4,IY) = V1
GO TO 97
96 UL = XP(3,IX)
CT = XP(4,IX) = XP(3,IX)
V1 = XP(1,IY)
VL = XP(3,IY) = V1
97 CONTINUE
C
C
C
C ESTABLISH SUB-REGION COUNTERS
IF (ISR(1) .GT. 0) GO TO 210
NS = NSREG
DO 200 I = 1,NS
200 ISR(I) = I
GO TO 230
210 NS = 0
```

CFINPT

CFINPT

```
DO 220 I = 1,10
IF (ISR(I) .LE. 0) GO TO 230
220 NS = NS + 1
C
230 IF (NS .LE. NSREG) GO TO 250
WRITE(TAPEOT,240) NS,NSREG
240 FORMAT(1H0, 8MSURFACE INTERPOLATION. THE ,I2,13H SUB-REGIONS ,
1 36HREQUEST 0 IS GREATER THAN VAILABLE./1H0, 12HCOUNTER SET ,
2 11HTO NSREG: , I2, 20H AND L. IE CONTYNUES,.)
C
NS = NSREG
250 CONTINUE
C
REWIND TAPEE
INTAPE = TAPEC
ITAPES = TAPED
C
I = 0
ISFR = 0
ICYCLE = 0
NYOT = 0
IF (IPF .NE. 0) GO TO 1520
1100 CONTINUE
ERROR = 0
CALL SFNTR3
IF (ERROR .EQ. 1) GO TO 1520
C
C START CYCLE ON THE ELEMENTS
1110 REWIND INTAPE
REWIND ITAPES
ISAVET = ITAPES
ICYCLE = ICYCLE + 1
IF (IPRINT .NE. 1) GO TO 1111
WRITE(TAPEOT,9030) INTAPE,ITAPES,LTJY
```

CFIN 246
CFIN 247
CFIN 248
CFIN 249
CFIN 250
CPIN 251
CFIN 252
CFIN 253
CFIN 254
CFIN 255
CFIN 256
CFIN 257
CFIN 258
CFIN 259
CFIN 260
CFIN 261
CFIN 262
CFIN 263
CFIN 264
CFIN 265
CFIN 266
CFIN 267
CFIN 268
CFIN 269
CFIN 270
CFIN 271
CFIN 272
CFIN 273
CFIN 274
CFIN 275
CFIN 276
CFIN 277
CFIN 278
CFIN 279
CFIN 280

CFINPT

```

9030 FORMAT(1M0,23HAT LOOP 1500; INTAPE = ,13; 5X9HITAPES = ,13.
1 5X7HLTOT = ,15)
1111 CONTINUE
DO 1500 II = 1,LTOT
IF (ICYCLE .GT. 1) GO TO 1050
C READ(TAPEE,END=1510) IG4,ELEM
READ(TAPEE) IG4,FLEM
IF (EOF(TAPEF)) 1510,1012
1012 LCOND(2) = ELEM(1)
GO TO 1090
1050 CONTINUE
C READ(INTAPE,END=1510) IG4,FDATA
READ(INTAPE) IG4,FDATA
IF (EOF(INTAPE)) 1510,1080
C
C CHECK IF ELEMENT ALREADY PROCESSED ON PREVIOUS CYCLE
1080 IF (LCOND(1) .NE. 0) GO TO 1200
1090 CONTINUE
IF (ISYM .EQ. 1) GO TO 1095
C REFLECT ELEMENT TO -Y SIDE AND CHANGE ORDER OF POINTS 2 AND 4
IG4 = -IG4
ELEM(5) = -ELEM(5)
ELEM(8) = -ELEM(8)
ELEM(15) = -ELEM(15)
ELEM(16) = -ELEM(16)
ELEM(17) = -ELEM(17)
ELEM(18) = -ELEM(18)
ELT = ELEM(12)
ELEM(12) = ELEM(14)
ELEM(14) = ELT
ELT = ELEM(16)
ELEM(16) = ELEM(18)
ELEM(18) = ELT
ELT = ELEM(20)
ELEM(20) = ELEM(22)

```

CFIN 281
CFIN 282
CFIN 283
CFIN 284
CFIN 285
CFIN 286I
CFIN 287C
CFIN 288C
CFIN 289
CFIN 290
CFIN 291
CFIN 292I
CFIN 293C
CFIN 294C
CFIN 295
CFIN 296
CFIN 297
CFIN 298
CFIN 299
CFIN 300
CFIN 301
CFIN 302
CFIN 303
CFIN 304
CFIN 305
CFIN 306
CFIN 307
CFIN 308
CFIN 309
CFIN 310
CFIN 311
CFIN 312
CFIN 313
CFIN 314
CFIN 315

CFINPT

```

ELEM(22) = ELT
C INTERPOLATE FOR SURFACE PROPERTIES AT CENTROIDS.
  1095 CONTINUE
  IC = 0
  IT = 1
  CALL VALU3(1, ELEM(7), ELEM(8), ELEM(9), XM, YN, ZN, IC, IT, IMP,
  1  DINF(1), DINF(2), DINF(3), DINF(4), DINF(5), DINF(6))
C
  LCOND(1) = 0
  IF (IMR.EQ. 0) GO TO 1200
  LCOND(1) = 1
  NTOT = NTOT + 1
C
C SAVE RESULTS ON TEMPORARY UNIT
  1200 CONTINUE
  WRITE(UNITAPES) IG4,FOATA
C
C GO TO NEXT ELEMENT
  1500 CONTINUE
  1510 CONTINUE
C
C 1520 I = I + 1
  IF (I.GT. 5) GO TO 1600
  ISFR = ISF(I)
  IF (IPRINT.NE. 1) GO TO 1525
  WRITE(TAPE0T,9010) I,ISFR,ICYCLE,NTOT,FROR
  9010 FORMAT(1H0, 4H1 = ,I3, 17HISFR = ,I3, 5X9MICYCLE = ,I3,
  1 7HNTOT = ,I5, 5X8ERRUR = ,I3)
  1525 CONTINUE
  IF (ISFR.NE. 0) GO TO 1530
  IF (IVIS.EQ. 1) GO TO 1520
  GO TO 1600
  1530 CONTINUE

```

CFINPT

CFINPT

```

IF (ICYCLE .EQ. 0) GO TO 1100
IF (NTOT .EQ. LTOT) GO TO 1600

C SWITCH TAPES (ONLY IF ERROR = )
IF (ERROR .NE. 0) GO TO 1100
II = ITAPES
ITAPES = INTAPE
INTAPE = II
GO TO 1100

C SAVE RESULTS ON TAPEF
1600 CONTINUE
IF (ICYCLE .GT. 0) GO TO 1560
WRITE(TAPEOT,1550)
1550 FORMAT(1H1,47HSURFACE INTERPOLATION (CFINT) CANNOT BE MADE./
154HOCHECK THAT PROPER FLOW FIELD DATA HAS BEEN SPECIFIED.,
2 16H RUN IS STOPPED.)
STOP

C 1560 REWIND ISAVET
IF (IPRINT .NE. 1) GO TO 1601
WRITE(TAPEOT,9020) LTOT,INTAPE,ISAVET
9020 FORMAT(1H0, 7H1TOT = ,15, 5X9HINTAPE = ,13, 5X9HISAVET = ,13)
1601 CONTINUE

DO 1610 L = 1,LTOT
READ(ISAVET,FND=1620) IG4,FDATA
READ(ISAVET) IG4,FDATA
IF (EOF(ISAVET)) 1620,1605
1605 WRITE(TAPEF) IG4,FDATA
ICT = ICT + 1
IF (IPRINT .NE. 1) GO TO 1610
WRITE(TAPEOT,9000) IG4,LCUND,ELEM,D.INFL
9000 FORMAT(1H0, 6HIG4 = ,15, 5X6HLCUND = , 313/
1 9H ELEM = , 10E12.4/1H , 7X,10E12.4/1H . 7X,5E12.4/
2 9H DINF = , 6E12.4)

```

CFIN 351
CFIN 352
CFIN 353
CFIN 354
CFIN 355
CFIN 356
CFIN 357
CFIN 358
CFIN 359
CFIN 360
CFIN 361
CFIN 362
CFIN 363
CFIN 364
CFIN 365
CFIN 366
CFIN 367
CFIN 368
CFIN 369
CFIN 370
CFIN 371
CFIN 372
CFIN 373
CFIN 374
CFIN 375
CFIN 376
CFIN 377
CFIN 378
CFIN 379
CFIN 380
CFIN 381
CFIN 382
CFIN 383
CFIN 384
CFIN 385

CFINPT

CFINPT

CFIN 386
CFIN 387
CFIN 388
CFIN 389
CFIN 390
CFIN 391
CFIN 392
CFIN 393
CFIN 394
CFIN 395
CFIN 396
CFIN 397
CFIN 398
CFIN 399
CFIN 400
CFIN 401
CFIN 402
CFIN 403

```
1610 CONTINUE
1620 CONTINUE
C CHECK FOR SYMMETRY REQUIREMENTS
  IF (8YMPCT.EQ.1 .OR. BETAS.EQ.0.0 .OR. ISYM.EQ.2) GO TO 1900
  REWIND TAPE
  ISYM = 2
  GO TO 10
C 1900 LTOTAB(IAB) = ICT
C END OF ALPHA=BETA LOOP
2000 CONTINUE
C REWIND TAPE
  REWIND TAPE
C RETURN
  END
```

CFINPT

INTEG

```

C C C C C
C THIS ROUTINE IS THE MAIN CONTROL ROUTINE FOR THE INTEGRAL BOUNDARY
C LAYER PROGRAM
C
C
SUBROUTINE INTFG
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH, ALT, PSTAG, TSTAG, V, REHO, PFS, TFS, AFS, RHOFS, VIS
COMMON /TAPE/TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /GASD/GAM, GASCOP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
INTEGER TAPFIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEF, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /GDATA/LTOT, J, SYMFCT, IORN, IGTYPF, L
COMMON /ABDATA/NAB, ALPHA(20), BETA(20), ROL(20), CDELTA(20), GI(20),
1 RI(20), PI(20)
COMMON /REF/SREF, MAC, SPAN, XCG, YCG, ZCG
COMMON /FSBS/FS(8), BS(8)
COMMON /CI/ R, PTZ, YTZ, UPMACH, NST, NVP, NTURB, KPVM, KEN, KSMTH,
1 KSPLN, KLE, KATCH, CTHT, DLAM, TLAM, DTURB, TTURB, KPFE, KGRAD, KSDE, KLAM,
2 KMAIN, KPROF, X(100), Y(100), PRES(100), UE(100), ME(100), POPTZ(100),
3 VOYCR(100), TWAL(100),
4 PSZ, YSZ, UZ, ASZ, ATZ, RHSZ, RHYZ, MUSZ, MUTZ, NUSZ, NUTZ, CP,
5 SPR, TC, ARCL,
7 S(100), SOL(100), AE(100), TSE(100),
8 TAWL(100), TAWT(100), TBAR(100), RM(300), SW(100), SUTHL(100),
9 RHSW(100), RHSE(100), HEADW(100), HEADE(100), NUW(100), MUBAR(100),
AAA(100), BB(100), FF(100), DUOS(100), DMUS(100), OMDL(100),
B THET(100), DELSR(100), DELTA(100), FORM(100),
C FORMI(100), FORMTR(100), RTH(100), RTHI(100), CF(100),
D YAUW(100), NUSS(100), DTDY(100), HTRAN(100), CRN(100),
E SHAPL(100), SHAPK(100), B2, NS,
F FTRAN, FORMS,
G INST, ITRAN, ISEP,
H XTAB(103), YTAB1(103), YTAB2(103), NTAB,

```

```

INTE 001
INTE 002
INTE 003
INTE 004
INTE 005
INTE 006
INTE 007
INTE 008
INTE 009
INTE 010
INTE 011
INTE 012
INTE 013
INTE 014
INTE 015
INTE 016
INTE 017
INTE 018
INTE 019
INTE 020
INTE 021
INTE 022
INTE 023
INTE 024
INTE 025
INTE 026
INTE 027
INTE 028
INTE 029
INTE 030
INTE 031
INTE 032
INTE 033
INTE 034
INTE 035

```

INTEG

INTEG

```

I      ERROR, TRANS, SEPRN
C      DIMENSION  TR(10), RE(2), TS(2), RT(2), TWAT(100), ILOC(100)
C      DIMENSION  LORG(20), CMU(20), CPR(20), CTC(20), CFLT(2)
C      DIMENSION  TITLE(15), ISF(5), Z(100)
C      DIMENSION  TITLEM(10), IMTAB(9), LOAB(20), TITLES(10), TITLEA(10),
1  TITLER(10), EA(12), E2(13), DATA(12), DATB(25), IDTYP(5),
2  LOFF(5), IFC(5), E3(17), ISTR(10), DATAS(25)
C
C      REAL MUSZ, MUTZ, MUSZ, MUTZ, MUSLE, MUSLM, MF, NUM, MUBAR
C      LOGICAL ERROR, TRANS, SEPRN
C      INTEGER ERROR, SYMFCT, PAGE, CASE
C
C      REAL  MACH, MAC
C
C      DATA FOR MU, PR, AND TC CURVE FITS
C
C      DATA CMU(1), CMU(2), CMU(3), CMU(4), CMU(5),
1  .0194517, .30175, .345113, .0682778, .0056659/
C      DATA CPR(1), CPR(2), CPR(3), CPR(4), CPR(5),
1  .R557, .234136, .1078624, .0236214, .0020296/
C      DATA CTC(1), CTC(2), CTC(3), CTC(4), CTC(5),
1  -.0383932, 1.269743, .3091125, 0.087438, .0094747/
C
C      WRITE (TAPEOT, 5)
C      5  FORMAT (I11, 4HINTEGRAL METHOD BOUNDARY LAYER DATA WILL BE ,
1  11HCALCULATED. )
C
C      INPUT REGION ID INFO AND NORMALIZATION FLAGS
C      10 READ (TAPEIN, 20) LASTR, NDSFT, IABSET, IR
C      20  FORMAT (I1, 3I2, 3I1)
C

```

```

INTE 036
INTE 037
INTE 038
INTE 039
INTE 040
INTE 041
INTE 042
INTE 043
INTE 044
INTE 045
INTE 046
INTE 047
INTE 048
INTE 049
INTE 050
INTE 051
INTE 052
INTE 053
INTE 054
INTE 055
INTE 056
INTE 057
INTE 058
INTE 059
INTE 060
INTE 061
INTE 062
INTE 063
INTE 064
INTE 065
INTE 066
INTE 067
INTE 068
INTE 069
INTE 070

```

INTEG

INTEG

```
WRITE (TAPEOT,32) LASTR,NDSET,IAHSET,IR
32  FORMAT (1H0,5X,6HLASTR,12,4X,6HNDSET,12,4X,7HIAHSET,12,4X,
1 4HIR,12)
C
C
C  READ IN MASTER DIRECTORY
   ITAG10 = 1
   CALL READMS (10,TITLEM,10,ITAG10)
   READ(10,ITAG10) TITLEM
   ITAG10 = 2
   CALL READMS (10,IMTAB,9,ITAG10)
   READ(10,ITAG10) IMTAB
   NEXT = IMTAB(2)
C
C  CHECK DATA SET NUMBER
   IF (IMTAB(1) .GE. NDSET) GO TO 40
   WRITE (TAPEOT,39) NDSET,IMTAB(1)
39  FORMAT (1H0,9H**NDSET =,12,31H IS GREATER THAN THE NUMBER OF ,
1 4HDATA SETS ACTUALLY ON UNIT 10. PROGRAM HALT. )
   STOP
C
C
C  READ IN FLOW DATA SET DIRECTORY
40  ITAG10 = IMTAB(NDSET + 4)
   CALL RFADMS (10,E4,12,ITAG10)
   READ (10,ITAG10) E4
   DO 41 I=1,10
41  TTILES(I) = F4(I)
      FMACH = E4(11)
      NARS = F4(12)
   ITAG10 = ITAG10 + 1
   CALL READMS (10,LOAD,20,ITAG10)
   READ(10,ITAG10) LOAD
C
C  READ IN FLOW REGION DIRECTORY FOR REQUESTED ALPHA=BETA SET
```

INTE 071
INTE 072
INTE 073
INTE 074
INTE 075
INTE 076
INTE 077
INTE 078C
INTE 079I
INTE 080
INTE 081C
INTE 082I
INTE 083
INTE 084
INTE 085
INTE 086
INTE 087
INTE 088
INTE 089
INTE 090
INTE 091
INTE 092
INTE 093
INTE 094
INTE 095C
INTE 096I
INTE 097
INTE 098
INTE 099
INTE 100
INTE 101
INTE 102C
INTE 103I
INTE 104
INTE 105

INTEG

INTEG

INTE 106
INTE 107C
INTE 106I
INTE 109
INTE 110
INTE 111
INTE 112
INTE 113
INTE 114
INTE 115C
INTE 116I
INTE 117
INTE 118
INTE 119
INTE 120
INTE 121C
INTE 122I
INTE 123
INTE 124
INTE 125
INTE 126
INTE 127
INTE 128
INTE 129
INTE 130
INTE 131
INTE 132
INTE 133
INTE 134
INTE 135
INTE 136
INTE 137
INTE 138
INTE 139
INTE 140

```
ITAG10 = LOAR(IARSET)
CALL READMS (10,E2,13,ITAG10)
READ (10:ITAG10) E2
DO 42 I=1,10
42 YTLFA(I) = E2(I)
ALPHAS = E2(11)
PETAS = E2(12)
NREG = E2(13)
ITAG10 = ITAG10 + 1
CALL READMS (10,LORG,20,ITAG10)
READ(10:ITAG10) LORG

C
C
C
C READ IN REQUESTED FLOW REGIONS
ITAG10 = LORG(CIR)
CALL READMS (10,E3,17,ITAG10)
READ (10:ITAG10) E3
DO 91 I=1,10
91 YTLER(I) = E3(I)
DO 92 I=1,5
IDTYP(I) = E3(I+10)
92 CONTINUE
IFTYP = IDTYP(1)
C CHECK IF PROPER STREAKLINE DATA ARE PRESENT
IF (IFTYP.EQ.3 .OR. IFTYP.EQ.4) GO TO 93
WRITE (TAPEOT,95)
95 FORMAT (1H,41M***STREAKLINE DATA ARE NOT ON UNIT 10 AS ,
1 24HREQUIRED. PROGRAM STOP. )
93 IFLOW = IDTYP(2)
IVIS = 0
IF (IFTYP .EQ. 4) IVIS = 1
94 NSREG = E3(16)
C
C
C
```

INTEG

INTEG

INTE 141
INTE 142
INTE 143
INTE 144
INTE 145
INTE 146
INTE 147
INTE 148
INTE 149
INTE 150
INTE 151
INTE 152
INTE 153
INTE 154
INTE 155
INTE 156
INTE 157
INTE 158
INTE 159
INTE 160
INTE 161
INTE 162
INTE 163
INTE 164
INTE 165
INTE 166
INTE 167
INTE 168
INTE 169
INTE 170
INTE 171
INTE 172
INTE 173
INTE 174
INTE 175

```
C C READ SKIN FRICTION METHOD CARD
  READ (TAPEIN,100) ISFM,INT,IPRINT,SURF16,SURF17,RETRAN
 100 FORMAT (3I1,7X,3F10.0)
  REFRAN = RETRAN * 1.0E+06

C C CHECK IF INTEGRAL METHOD IS TO BE USED FOR SKIN FRICTION
  IF (ISFM.EQ.1) GO TO 162

C C
C C INITIALIZE BOUNDARY LAYER CONSTANTS
  R = GASCP*(GAM=1.0)/GAM
  TTOT = 1.0 + (GAM=1.0)/2.0*MACH*MACH
  TT7 = TFS * TTOT
  PTZ = PFS * TTOT**((GAM/(GAM=1.0)))
  UPMACH = MACH
  KCM = 0

C C INITIALIZE STATIC AND TOTAL PARAMETERS
  TSLE = 518.688
  TSLM = 288.160
  MUSLE = 3.711402E=7
  MUSLM = 1.777029E=5
  TCSLE = 3.202206E=3
  TCSLM = 2.561796E=2
  TSZ = TFS
  PSZ = PFS
  RSHZ = PSZ/R/TSZ
  RHTZ = PTZ/R/TTZ
  ASZ = SORT(GAM*R*TSZ)
  ATZ = SORT(GAM*R*TTZ)
  UZ = UPMACH*ASZ
  CP = G=5CP
  IF (MEM.EQ.1) GO TO 110
```

INTEG

INTEG

INTE 176
INTE 177
INTE 178
INTE 179
INTE 180
INTE 181
INTE 182
INTE 183
INTE 184
INTE 185
INTE 186
INTE 187
INTE 188
INTE 189
INTE 190
INTE 191
INTE 192
INTE 193
INTE 194
INTE 195
INTE 196
INTE 197
INTE 198
INTE 199
INTE 200
INTE 201
INTE 202
INTE 203
INTE 204
INTE 205
INTE 206
INTE 207
INTE 208
INTE 209
INTE 210

INTEG

```

TCO# 198,60
TR1# TSZ/TSLE
TR2# TYZ/TSLE
GO TO 120
110 TCON# 110,33
TR1# TSZ/TSLM
TR2# TYZ/TSLM
120 CALL CURVFT(CPR,PR,TR1,0,4,0)
CALL CURVFT(CTC,TC,TR1,0,4,0)
CALL CURVFT(CMU,MUSZ,TR1,0,4,0)
CALL CURVFT(CMU,MUTZ,TR2,0,4,0)
IF (KEM,EQ,1) GO TO 130
TC# TC*TC$LE
MUSZ# MUSZ*MUSLE
MUTZ# MUTZ*MUSLE
GO TO 140
130 TC# TC*TC$LM
MUSZ# MUSZ*MUSLM
MUTZ# MUTZ*MUSLM
140 MUSZ# MUSZ/RHSZ
MUTZ# MUTZ/RHTZ
C
C
C
C
READ INTEGRAL METHOD CONTROL CARD
READ (TAPEIN,141) NVP,NTURB,KPVH,#SMTH,K$PLN,KLE,KATCH,KPRE,
1 KGRAD,KSDE,KLAM,KMAIN,KPROF,CTHET,DLAM,TLAM,DTURB,TTURB
141 FORMAT (2I2,I1,I2,9I1,4X,5F10,0)
C
WRITE (TAPEOT,142) NVP,NTURB,KPVM,K$MTH,K$PLN,KLE,KATCH,KPRE,
1 KGRAD,KSDE,KLAM,KMAIN,KPROF,CTHET,DLAM,TLAM,DTURB,TTURB
142 FORMAT (1H0,28HINTEGRAL METHOD CONTROL DATA,/1H ,4HNVP#,I3,3X,
1 6HNTURB#,I3,3X,5HKPV##,I3,3X,6HKSMT##,I3,3X,6HKSPLN#,I3,3X,
2 4HKLE#,I3,3X,6HKAATCH#,I3,3X,5HKPRE#,#I1, /1H ,6HKGRADE#,I3,3X,
3 5HKSDE#,I3,3X,5HKLAMB#,I3,3X,6HKMAINS#,I3,3X,6HKPROF#=#,I3, /1H ,

```

INTEG

```

4 6HC7HETB,F10.5,3X,5HDLAMB,F10.5,3X,5HTLAMB,F10.5,3X,6HOTURRB=,
5 F10.5,3X,6HTTURB=F10.5)
WRITE (TAPEOT,143) PZ,TSZ,UZ,ASZ,ATZ,PMSZ,RMTZ,MUSZ,MUTZ,MUSZ,
1 NUTZ,CP,TC
143 FORMAT (I40,16HBASIC PARAMETERS,/1H ,3X,6MPSZ #,F13.5,3X,
1 6MTSZ #,F13.5,3X,6MUZ #,F13.5,3X,6HASZ #,F13.5,3X,6MATZ #,
2 F13.5,/1H ,3X,6MRSZ #,E13.5,3X,6MRTZ #,E13.5,3X,6HMUSZ #,
3 F13.5,3X,6MNUTZ #,E13.5,3X,6MNSZ #,F13.5,/1H ,3X,6MNUZ #,
4 F13.5,3X,6MCP #,F13.5,3X,6MTC #,F13.5)
162 CONTINUE
C
C READ STREAMLINE SFLGCTIION CARD
150 READ (TAPEIN,160) ISTR
160 FORMAT (10I2)
WRITE (TAPEOT,144) ISTR
144 FORMAT (I40, 20HSTREAMLINES TO BE ANALYZED . 1014 )
C
C START STREAMLINE DO LOOP
DO 620 N=1,10
IF (JISTR(N) .EQ. 0) GO TO 630
ISRS = ISTR(N)
WRITE (TAPEOT,161) ISRS
161 FORMAT (I40,20HSTREAMLINE BEING ANALYZED #,I3)
ISTRAN = 0
ITRAN = 0
ISEP = 0
C CHECK IF STREAMLINE IS ON UNIT 10
IF (JSRS .LE. NSREG) GO TO 170
WRITE (TAPEOT,165)
165 FORMAT (I40,47H***REQUESTED STREAMLINE NUMBER IS GREATER THAN ,
1 56HTHE NUMBER OF STREAMLINES ACTUALLY STORED ON UNIT 10. ,

```

```

INTE 211
INTE 212
INTE 213
INTE 214
INTE 215
INTE 216
INTE 217
INTE 218
INTE 219
INTE 220
INTE 221
INTE 222
INTE 223
INTE 224
INTE 225
INTE 226
INTE 227
INTE 228
INTE 229
INTE 230
INTE 231
INTE 232
INTE 233
INTE 234
INTE 235
INTE 236
INTE 237
INTE 238
INTE 239
INTE 240
INTE 241
INTE 242
INTE 243
INTE 244
INTE 245

```

INTEG

INTEG

INTE 246
INTE 247
INTE 248
INTE 249
INTE 250C
INTE 251I
INTE 252
INTE 253
INTE 254
INTE 255
INTE 256
INTE 257
INTE 258
INTE 259
INTE 260
INTE 261
INTE 262
INTE 263
INTE 264
INTE 265
INTE 266
INTE 267
INTE 268
INTE 269
INTE 270C
INTE 271I
INTE 272
INTE 273
INTE 274
INTE 275
INTE 276
INTE 277
INTE 278
INTE 279
INTE 280

```
      2 13PROGRAM STOP. )  
170 CONTINUE  
C  
      ITAG10 = LORG(IR) + ISRS + 4  
      CALL READMS (10,DATA,25,ITAG10)  
      READ (10:ITAG10) DATA  
      DO 180 I=1,5  
        LOFF(I) = DATA(I)  
180  IFC(I) = DATA(I+5)  
C  
      GFT STREAMLINE COORDINATES AND PRESSURE DATA FROM UNIT 10  
C  
      NUMBER OF DATA POINTS ON UNIT 10 FOR THIS STREAMLINE  
      NST = IFC(1)  
      IF (NST .GT. 0) GO TO 182  
      WRITE (TAPEOT,181)  
181  FORMAT (1H0,34M***THERE ARE 40 STREAMLINES ON UNIT 10. STOP )  
      STOP  
182 ITAG10 = LOFF(1)  
C*****  
C  READ IN STREAMLINE DATA POINTS  
      GAMR = GAM * R  
      ITRC = 0  
      DO 190 I=1,NST  
        CALL READMS (10,DATA,12,ITAG10)  
        READ (10:ITAG10) DATA  
        ILDC(I) = ITAG10  
        ITAG10 = ITAG10 + 1  
C  
      X(I) = DATA(1)  
      Y(I) = DATA(2)  
      Z(I) = DATA(3)  
      S(I) = DATA(4)  
      ME(I) = DATA(7)  
      PRES(I) = DATA(11)*PFS
```

INTEG

INTEG

```

YSE(I) = DATA(12)*YFS
AE(I) = SORT(GAMR*YSE(I))
UE(I) = ME(I) * AE(I)
C
C CHECK IF INPUT WALL TEMP IS TO BE USED
IF (ISFM .EQ. 1) GO TO 183
IF (IMT .EQ. 2) GO TO 186
IF (IMT .EQ. 5) GO TO 188
IF (IMT .EQ. 7) GO TO 189
IF (IMT .EQ. 9) GO TO 189
C
C 183 CONTINUE
RS(7) = UE(I)
RS(3) = YSE(I)
RS(1) = PRES(I)/(R*YSE(I))
ZL = S(I)
CALL TEMP (EL,TR,RE,TS,IMT,MER,IPRINT,RT,CFLT)
TWAL(I) = TR(5)
TWAT(I) = TR(6)
C SET CF ARRAY TO VALUES CALCULATED BY TEMP ROUTINE
IF (RE(1) .GT. RETRAN) GO TO 202
CF(I) = CFLT(1)
GO TO 190
202 IF (ITRC .EQ. 1) GO TO 203
ITRAN = I
ITRC = 1
203 CF(I) = CFLT(2)
GO TO 190
C
188 TWAL(I) = SURF16
190 CONTINUE
C
IF (ISFM .EQ. 1) GO TO 495
C
```

INTE 281
INTE 282
INTE 283
INTE 284
INTE 285
INTE 286
INTE 287
INTE 288
INTE 289
INTE 290
INTE 291
INTE 292
INTE 293
INTE 294
INTE 295
INTE 296
INTE 297
INTE 298
INTE 299
INTE 300
INTE 301
INTE 302
INTE 303
INTE 304
INTE 305
INTE 306
INTE 307
INTE 308
INTE 309
INTE 310
INTE 311
INTE 312
INTE 313
INTE 314
INTE 315

INTEG

INTEG

316 INTE
 317 INTE
 318 INTE
 319 INTE
 320 INTE
 321 INTE
 322 INTE
 323 INTE
 324 INTE
 325 INTE
 326 INTE
 327 INTE
 328 INTE
 329 INTE
 330 INTE
 331 INTE
 332 INTE
 333 INTE
 334 INTE
 335 INTE
 336 INTE
 337 INTE
 338 INTE
 339 INTE
 340 INTE
 341 INTE
 342 INTE
 343 INTE
 344 INTE
 345 INTE
 346 INTE
 347 INTE
 348 INTE
 349 INTE
 350 INTE

INTEG

```

C PRELIMINARY CALCULATIONS
  ERROR = .FALSE.
  TRANS = .FALSE.
  SEPRN = .FALSE.
  CALL PRECAL
  IF (ERROR) GO TO 1000

C LAMINAR CALCULATIONS
  CALL LAMNAR
  IF (ERROR) GO TO 1010
  IF (SEPRN) GO TO 490
  IF (.NOT. TRANS) GO TO 490

C TURBULENT CALCULATIONS
  IF (KEM .EQ. 1) GO TO 210
  TCON = 198.6
  GO TO 215
210 TCON = 110.33
215 CONTINUE
  IF (IMT .EQ. 2) GO TO 221
  IF (IMT .EQ. 5) GO TO 221
  IF (IMT .EQ. 7) GO TO 221
  IF (IMT .EQ. 4) GO TO 221
  DO 220 I=1, NST
    TWAL(I) = TWAL(I)
    TEM1 = 1. + .5*(GAM-1.)*ME(I)**2
    RMSW(I) = PRES(I)/R/TWAL(I)
    RMSE(I) = PRES(I)/R/TSE(I)
    HEADW(I) = .5*RHSW(I)*UE(I)**2
    HEADF(I) = .5*RHSE(I)*UE(I)**2
    SW(I) = TWAL(I)/TTZ-1.
    SUTWL(I) = SORT(TWAL(I)/TTZ)*(TTZ+TCON)/(TWAL(I)+TCON)
    NUW(I) = SUTWL(I)*(1.+SW(I))*NUTZ*RMTZ/RHSW(I)
    RW(I) = UE(I)*S(I)/NUW(I)
  
```

INTEG

331 INTE
 332 INTE
 333 INTE
 334 INTE
 335 INTE
 336 INTE
 337 INTE
 338 INTE
 339 INTE
 360 INTE
 361 INTE
 362 INTE
 363 INTE
 364 INTE
 365 INTE
 366 INTE
 367 INTE
 368 INTE
 369 INTE
 370 INTE
 371 INTE
 372 INTE
 373 INTE
 374 INTE
 375 INTE
 376 INTE
 377 INTE
 378 INTE
 379 INTE
 380 INTE
 381 INTE
 382 INTE
 383 INTE
 384 INTE
 385 INTE

```

TAWL(I) = TSE(I)*(1.+PR*(1./2.))* (TEM1*(1.))
TAWT(I) = TSE(I)*(1.+PR*(1./3.))* (TEM1*(1.))
TBAR(I) = .5*(TAWL(I)+TSE(I))+.22*PR*(1./3.)*(TYZ+TSE(I))
MURAR(I) = MUTZ*STHML(I)+TBAR(I)/TYZ
BB(I) = UE(I)*RHSE(I)/MUTZ
AAA(I) = BB(I)*TSE(I)/TBAR(I)*(MUBAR(I)/MUTZ)**.268
PF(I) = 1.+1509*ME(I)**2+.60*SW(I)+.2101*SW(I)*ME(I)**2+.0114*ME(I)
1**4+.0180*SW(I)*ME(I)**4+.1825*SW(I)**2+.0735*SW(I)**2*ME(I)**2
2+.0073*SW(I)**2*ME(I)**4
220 CONTINUE
221 CONTINUE
    CALL TURBLN
    IF (ERCR) GO TO 1020
490 CONTINUE
    IDSOUT = 0
    LU = 1
    CALL PROFIL (IDSOUT,LU)
C
C 495 CONTINUE
C
C SET UP AND SAVE CF DATA BACK ON UNIT 10
C INITIALIZE COUNTERS FOR DATA SAVE LOGIC
C WHEN TRANSITIONAL CAPABILITY IS INCLUDED. REMOVE THE NEXT CARD
    ISTRAN = 0
    IF (ISTRAN .LE. 0) ISTRAN = 9999
    IF (ITRAN .LE. 0) ITRAN = 9999
    IF (ISEP .LE. 0) ISEP = 9997
    IFC1 = 0
    IFC2 = 0
    IFC3 = 0
    IFC4 = 0
    LOFF1 = 0
    LOFF2 = 0
    LOFF3 = 0
    LOFF4 = 0
  
```

INTEG

INTEG

```

      ITAG10 = LOFF(1)
      IF (CF(1) .LE. 0.0 .AND. ISEP .NE. 1)
1     CF(1) = EXP(ALOG(CF(2))) + (ALOG(CF(2)) - ALOG(CF(3)))
2     *(S(2) - S(1))/(S(3) - S(2))
C
C   ON LOOP TO STORE SKIN FRICTION FOR ALL DATA POINTS ON STREAMLINE
      DO 600 I=1,MST
C
C   CHECK STATUS OF POINT (LAMINAR, TRANSITIONAL, TURBULENT, OR SEPARATED)
C   CHECK IF NOT SEPARATED
      IF (I .LT. ISEP) GO TO 520
C   SEPARATED FLOW
      IF (I .NE. ISEP) GO TO 510
      LOFF4 = ITAG10
      IFC4 = 0
510    IFC4 = IFC4 + 1
      CF(I) = 0.0
      GO TO 580
C
C   CHECK IF TURBULENT
520    IF (I .LT. ITRAN) GO TO 540
      TURBULENT
      IF (I .NE. ITRAN) GO TO 530
      LOFF3 = ITAG10
      IFC3 = 0
530    IFC3 = IFC3 + 1
      GO TO 580
C
C   LAMINAR OR TRANSITIONAL
540    IF (I .LT. ISTRAN) GO TO 560
      TRANSITIONAL
      IF (I .NE. ISTRAN) GO TO 550
      LOFF2 = ITAG10
      IFC2 = 0
550    IFC2 = IFC2 + 1

```

INTE 386
INTE 387
INTE 388
INTE 389
INTE 390
INTE 391
INTE 392
INTE 393
INTE 394
INTE 395
INTE 396
INTE 397
INTE 398
INTE 399
INTE 400
INTE 401
INTE 402
INTE 403
INTE 404
INTE 405
INTE 406
INTE 407
INTE 408
INTE 409
INTE 410
INTE 411
INTE 412
INTE 413
INTE 414
INTE 415
INTE 416
INTE 417
INTE 418
INTE 419
INTE 420

INTEG

INTEG

```

      GO TO 580
C     LAMINAR
C     560 IF (I .NE. 1) GO TO 570
      LOFF1 = ITAG10
      IFC1 = 0
C     570 IFC1 = IFC1 + 1
C     580 CONTINUE
C     590 RECALL DATA
      CALL READMS (10,DATA,12,ITAG10)
      READ (10,ITAG10) DATA
C     600 SAVE SKIN FRICTION COEFFICIENT
      DATA(5) = CF(I)
      CALL WRITMS (10,DATA,12,ITAG10)
      WRITE (10,ITAG10) DATA
      WRITE (TAPEOT,900) IYAG10,DATA
C     900 FORMAT (1H , 7HITAG10=,15,6E12.5, /1H ,12X,6E12.5)
C     ITAG10 = ITAG10 + 1
C     600 CONTINUE
C     610 RESET REGION DIRECTORY COUNTERS
      DATB(21) = 0.0
      DATB(22) = 0.0
      DATB(23) = 0.0
      DATB(24) = 0.0
      DATB(25) = 0.0
      DO 611 I=1,25
C     611   DATBS(I) = 0.0
C     CHECK IF LAMINAR FLOW WAS FOUND

```

```

INTE 021
INTE 022
INTE 023
INTE 024
INTE 025
INTE 026
INTE 027
INTE 028
INTE 029
INTE 030
INTE 031
INTE 032C
INTE 033I
INTE 034
INTE 035
INTE 036
INTE 037C
INTE 038I
INTE 039
INTE 040
INTE 041
INTE 042
INTE 043
INTE 044
INTE 045
INTE 046
INTE 047
INTE 048
INTE 049
INTE 050
INTE 051
INTE 052
INTE 053
INTE 054
INTE 055

```

INTEG

INTEG

INTE 456
INTE 457
INTE 458
INTE 459
INTE 460
INTE 461C
INTE 462I
INTE 463
INTE 464
INTE 465
INTE 466
INTE 467
INTE 468
INTE 469
INTE 470
INTE 471C
INTE 472I
INTE 473
INTE 474
INTE 475
INTE 476
INTE 477
INTE 478
INTE 479
INTE 480
INTE 481C
INTE 482I
INTE 483
INTE 484
INTE 485
INTE 486
INTE 487
INTE 488
INTE 489
INTE 490

```
IF (IFC1 .EQ. 0) GO TO 612
DATB(21) = NEXT
DATBS(1) = LOFF1
DATBS(6) = IFC1
ITAG10 = NEXT
CALL WRITMS (10,DATBS,25,ITAG10)
WRITE (10,ITAG10) DATBS
NEXT = NEXT + 1

C
C CHECK IF TRANSITIONAL FLOW WAS FOUND
612 IF (IFC2 .EQ. 0) GO TO 614
DATB(22) = NEXT
DATBS(1) = LOFF2
DATBS(6) = IFC2
ITAG10 = NEXT
CALL WRITMS (10,DATBS,25,ITAG10)
WRITE (10,ITAG10) DATBS
NEXT = NEXT + 1

C
C CHECK IF TURBULENT FLOW WAS FOUND
614 IF (IFC3 .EQ. 0) GO TO 616
DATB(23) = NEXT
DATBS(1) = LOFF3
DATBS(6) = IFC3
ITAG10 = NEXT
CALL WRITMS (10,DATBS,25,ITAG10)
WRITE (10,ITAG10) DATBS
NEXT = NEXT + 1

C
C CHECK IF SEPARATED FLOW WAS FOUND
616 IF (IFC4 .EQ. 0) GO TO 618
DATB(24) = NEXT
DATBS(1) = LOFF4
DATBS(6) = IFC4
ITAG10 = NEXT
```

INTEG

INTEG

INTE 491C
 INTE 492I
 INTE 493
 INTE 494
 INTE 495
 INTE 496
 INTE 497C
 INTE 498I
 INTE 499
 INTE 500
 INTE 501
 INTE 502
 INTE 503
 INTE 504
 INTE 505
 INTE 506
 INTE 507
 INTE 508
 INTE 509C
 INTE 510I
 INTE 511
 INTE 512
 INTE 513
 INTE 514C
 INTE 515I
 INTE 516
 INTE 517
 INTE 518
 INTE 519
 INTE 520
 INTE 521
 INTE 522
 INTE 523
 INTE 524

```

CALL WRITMS (10,DAT89,25,ITAG10)
WRITE (10#ITAG10) DAT88
NEXT = NEXT + 1

C
C RESET REGION TABLE
618 ITAG10 = LORG(IR) + ISRS + 4
CALL WRITMS (10,DAT8,25,ITAG10)
WRITE (10#ITAG10) DAT8

C
C STREAMLINE COMPLETE
620 CONTINUE

C
C ALL STREAMLINES COMPLETED
630 CONTINUE

C
C RESET MASTER DIPECTORY
INTAB(2) = NEXT
ITAG10 = 2
CALL WRITMS (10,IMTAB,9,ITAG10)
WRITE (10#ITAG10) IMTAB

C
C CHANGE REGION DIRECTORY TO INDICATE THAT CF HAS BEEN SAVED ON IT
E3(11) = 4
ITAG10 = LORG(IR)
CALL WRITMS (10,E3,17,ITAG10)
WRITE (10#ITAG10) E3

C
C
C IF (LASTR .EQ. 0) GO TO 10
RETURN

C
1000 STOP
1010 STOP
1020 STOP
END
  
```

INTEG

PRECAL

```

001 PREC
002 PREC
003 PREC
004 PREC
005 PREC
006 PREC
007 PREC
008 PREC
009 PREC
010 PREC
011 PREC
012 PREC
013 PREC
014 PREC
015 PREC
016 PREC
017 PREC
018 PREC
019 PREC
020 PREC
021 PREC
022 PREC
023 PREC
024 PREC
025 PREC
026 PREC
027 PREC
028 PREC
029 PREC
030 PREC
031 PREC
032 PREC
033 PREC
034 PREC
035 PREC

```

```

SURROUTINE PRECAL
THIS ROUTINE WAS TAKEN FROM NASA TN D-5681 BY W.D. McNALLY,
MODIFIED BY A.E. GENTRY

COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH, ALY, PSTAG, YSTAG, V, PENO, PFS, TFS, AFS, RHOF, VIS
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /GASD/GAM, GASC, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON/C1/ R, PIZ, YTZ, UPWACH, NST, NVP, NTURB, KPVM, KEM, KSMTH,
      1KSPLN, KLE, KATCH, CTHEY, DLAM, TLAM, DTUFR, TTURR, KPRE, KGRAD, KSDF, KLAM,
      2KMAIN, KPROF, X(100), Y(100), PRES(100), UE(100), WE(100), POPTZ(100),
      3VOVCR(100), THAL(100),
      4PSZ, YSZ, UZ, ASZ, ATZ, RHSZ, RMTZ, MUSZ, MUTZ, NUSZ, CP,
      5PR, TC, ARCL,
      6
      7
      8TAWL(100), TANT(100), TBAR(100), RM(300), SM(100), SUTHL(100),
      9RHSW(100), RHSZ(100), HEADW(100), HEADE(100), NUW(100), MUBAR(100),
      AAAA(100), BB(100), FF(100), DUDS(100), DMD3(100), DMDL(100),
      10TMT(100), DELSR(100), DELTA(100), FGRM(100),
      11CFORMI(100), FORMTR(100), RTH(100), RTHI(100), CF(100),
      12DTAUW(100), MUSS(100), DTOY(100), HTRAN(100), CRN(100),
      13E SHAPL(100), SHAPK(100), B, NS,
      14F FTRAN, FORMS,
      15G INST, ITRAN, ISEP,
      16H XTAB(103), YTAB1(103), YTAB2(103), NTAB,
      17I EROR, TRANS, SEPRN
      DIMENSION SDCR(100)
      REAL MUSZ, MUTZ, NUSZ, MUSLE, MUJLM, ME, NUW, MUBAR, MACH
      LOGICAL EROR, TRANS, SEPRN

```

```

C
C
C
C
C
C

```

PRECAL

PRECAL

```

C
C  CALCULATE ARC LENGTH RATIOS
  ARCL = S(NST)
  DO 60 I=1,NST
    60 SOL(I) = S(I)/ARCL

C
C  CALCULATE POPTZ AND VOVCR AT EACH STATION
  GAM1 = (GAM+1.)/(GAM+1.)
  GAM2 = (GAM-1.)/GAM
  PTOTAL = PTZ
  VCR = SORT(2.0*GAM*R*TTZ/(GAM + 1.0))
  DO 80 I=1,NST
    POPTZ(I) = PRES(I) / PTOTAL
    VOVCR(I) = UE(I)/VCR
  80 CONTINUE

C
C  170 WRITE (TAPEOT,1000)
  WRITE (TAPEOT,1020) (I,PRES(I),UE(I),ME(I),POPTZ(I),VOVCR(I),
  1 I=1,NST)

C
C  SMOOTH INPUT DATA IF NECESSARY
  IF (KSMTH.LT.1) GO TO 200
  KDNF = 0
  GAMR = GAM * R
  180 CALL SMTHNA (ME,NST,0)
  CALL SMTHNA (PRES,NST,0)
  KDNNEKDONE=1
  RECOMPUTE PRES,ME,POPTZ, AND VOVCR AT EACH STATION
  DO 100 I=1,NST
    TSF(I) = TTZ/(1.0 + 0.5*(GAM - 1.0)*ME(I)**2)
    AE(I) = SORT(GAMR*TSF(I))
    UE(I) = ME(I) * AE(I)
  IF (I.EQ.1 .AND. UE(I).LT.0.0) UE(I) = 0.0
  IF (I.GT.1 .AND. UE(I).LT.0.0) UE(I) = 0.01*FLOAT(I)

```

PRECAL

PRECAL

PREC 071
PREC 072
PREC 073
PREC 074
PREC 075
PREC 076
PREC 077
PREC 078
PREC 079
PREC 080
PREC 081
PREC 082
PREC 083
PREC 084
PREC 085
PREC 086
PREC 087
PREC 088
PREC 089
PREC 090
PREC 091
PREC 092
PREC 093
PREC 094
PREC 095
PREC 096
PREC 097
PREC 098
PREC 099
PREC 100
PREC 101
PREC 102
PREC 103
PREC 104
PREC 105

```
POPTZ(I) = PRES(I) / POTAL
100 VOVCR(I) = UF(I)/VCR
190 WRITE (TAPEOT,1040)
WRITE (TAPEOT,1020) (I,PRES(I),UE(I),ME(I),POPTZ(I),VOVCR(I),
1 IBI,NST)
IF(KDONE.LT.KSMTH) GO TO 180
C PRINT GEOMETRY PARAMETERS
C
200 IF (KPRE.NE.1) GO TO 210
WRITE (TAPEOT,1030) (I,X(I),Y(I),S(I),SOL(I),IBI,NST)
C CALCULATE OTHER NECESSARY PARAMETERS AT EACH STATION
C
210 CONTINUE
C
IF (KEM.EQ. 1) GO TO 110
TCON = 198.6
GO TO 120
110 TCON = 110.33
120 CONTINUE
C
DO 220 IBI,NST
TEM1 = 1. + .5*(GAM-1.)*ME(I)**2
RHS*(I) = PRES(I)/R/TWAL(I)
RHSE(I) = PRES(I)/R/TSE(I)
HEADW(I) = .5*RHSW(I)*UE(Y)**2
HEADE(I) = .5*RHSF(I)*UE(Y)**2
SW(I) = TWAL(I)/TTZ*1.
SUTHL(I) = SQRT(TWAL(I)/TTZ)*(TTZ+YCON)/(TWAL(I)+TCON)
NUM(I) = SUTHL(I)*(1.+SW(I))*NUITZ*RHTZ/RHSW(I)
RW(I) = UE(I)*S(I)/NUM(I)
TAWL(I) = TSE(I)*(1.+PR**((1./2.)*TEM1-1.))
C
```

PRECAL

PRECAL

```

TAWT(I)*TSE(I)*(1.+PR*(1./3.))*(TEM1=1.)
TBARC(I)=.5*(TVAL(I)+TSE(I))+.22*PR*(1./3.)*(TTZ+TSE(I))
MURAR(I)=MUTZ*SUTHL(I)*TBAR(I)/TTZ
BB(I)=UE(I)*RHSE(I)/MUTZ
AAA(I)=BB(I)*TSE(I)/TRAR(I)*(MUBAR(I)/MUTZ)**.268
WRITE(6,9000) I,AAA(I),BB(I),I4,2F20.6)
C9000 FORMAT(26HIN PRECAL = I,AAA(I),BB(I),I4,2F20.6)
FF(I)=1.+1599*ME(I)**2+.60*SW(I)+.2101*SW(I)*ME(I)**2+.0114*ME(I)
1**4+.0180*SW(I)*ME(I)**4+.1925*SW(I)**2+.0735*SW(I)**2*ME(I)**2
2+.0773*SW(I)**2*ME(I)**4
220 CONTINUE
C
C COMPUTE VELOCITY AND MACH NUMBER GRADIENTS ALONG THE SURFACE
C
C FINITE DIFFERENCE TECHNIQUE
IF(KSPLN.EQ.1) GO TO 230
CALL GRADNT(S,UE,NST,DUDS)
CALL GRADNT(S,ME,NST,DMDS)
GO TO 240
C SPLINE CURVE TECHNIQUE
230 CALL SPLINE(S,UE,NST,DUDS,SDER)
CALL SPLINE(S,ME,NST,DMDS,SDER)
240 DO 250 I=1,NST
250 DMDL(I)=ARCL*DMDS(I)
C
C PRINT OTHER CALCULATED PARAMETERS
C
IF(KPRE.NE.1) GO TO 260
WRITE(TAPEOT,1050) (I,AE(I),TSE(I),TVAL(I),TAWL(I),TAWT(I),
1 TBAR(I),I=1,NST)
WRITE(TAPEOT,1060) (I,RW(I),SW(I),SUTHL(I),RHSHW(I),RHSE(I),
1 HEADW(I),HEADC(I),NUM(I),MUBAR(I),I=1,NST)
260 IF(KGRAD.NE.1) GO TO 270
WRITE(TAPEOT,1070)
WRITE(TAPEOT,1080) (I,DUDS(I),OMDS(I),OMDL(I),I=1,NST)

```

PREC 106
PREC 107
PREC 108
PREC 109
PREC 110
PREC 111
PREC 112
PREC 113
PREC 114
PREC 115
PREC 116
PREC 117
PREC 118
PREC 119
PREC 120
PREC 121
PREC 122
PREC 123
PREC 124
PREC 125
PREC 126
PREC 127
PREC 128
PREC 129
PREC 130
PREC 131
PREC 132
PREC 133
PREC 134
PREC 135
PREC 136
PREC 137
PREC 138
PREC 139
PREC 140

PRECAL

PRECAL

```

C CHECK FOR IMPROPER INPUT
C
C 270 DO 280 I=2,NST
    IF (UE(I).NE.0.) GO TO 280
    IF (I.EQ. NST) GO TO 280
    ERROR = TRUE
    WRITE (TAPE07,1090)
    RETURN
280 CONTINUE
    RETURN
290 ERROR = TRUE
    WRITE (TAPE07,1100)
    RETURN
C
C FORMAT STATEMENTS
C
1000 FORMAT(1H1//4X,24HPRELIMINARY CALCULATIONS//)
1020 FORMAT(1X,7HSTATION,7X,4HPRES,13X,2HUF,12X,2HME,11X,5HPOPTZ,9X,5H
1VOVCR/(2X,I3,5X,F12.5,3X,F12.5,4X,F10.6,4X,F10.6,4X,F10.6))
1030 FORMAT(//1X,7HSTATION,7X,1HX,12X,1HX,12X,1HX,
19X,3HSOL/(2X,I3,3X,F12.5,1X,F12.5,3X,F9.5
2))
1040 FORMAT(//1X,64HSMOOTHED SURFACE DISTRIBUTIONS OF PRES, UE, ME, PD
1PTZ, AND VOVCR)
1050 FORMAT(//1X,7HSTATION,5X,2HAE,10X,3HISE,9X,4HTAL,8X,4HTAL,8X,4H
1TAWT,8X,4HTBAR/(1X,I3,4X,F9.3,5(4X,F8,3))
1060 FORMAT(//1X,7HSTATION,11X,2HRM,6X,2HWS,4X,5HSUTHL,7X,4HRMSH,12X,4
1HRHSF,8X,5HHEADN,4X,5HHEAD,9X,3HNUM,12X,5HUBAR/(2X,I3,3X,F16.1,2
2X,F4.1,1X,F7.3,2X,614.6,2X,614.6,1X,F8.3,1X,F8.3,2X,614.6,2X,614.6
3))
1070 FORMAT(1H1//21X,17HSURFACE GRADIENTS./)
1080 FORMAT(1X,7HSTATION,13X,4HWDSD,15X,4HWDSD,15X,4HWDSD/(2X,I3,4X,F18
1.6,1X,F18.6,1X,F18.6))
1090 FORMAT(//10X,83HTHERE IS A STAGNATION POINT AT A STATION OTHER

```

```

PREC 141
PREC 142
PREC 143
PREC 144
PREC 145
PREC 146
PREC 147
PREC 148
PREC 149
PREC 150
PREC 151
PREC 152
PREC 153
PREC 154
PREC 155
PREC 156
PREC 157
PREC 158
PREC 159
PREC 160
PREC 161
PREC 162
PREC 163
PREC 164
PREC 165
PREC 166
PREC 167
PREC 168
PREC 169
PREC 170
PREC 171
PREC 172
PREC 173
PREC 174
PREC 175

```

PRECAL

PRECAL

```
1100 ITHAN STATION !. THIS IS NOT ALLOWED)
1100 FORMAT(////10X,11I10AN INPUT PRESSURE, VELOCITY, OR MACH NUMBER IS
1100 IF EITHER LESS THAN ZERO OR GREATER THAN ITS MAXIMUM ALLOWABLE VALUE)
1100 END
```

PREC 176
PREC 177
PREC 178
PREC 179

PRECAL

SFNTR3

```

C
C
      SURROUTINE SFNTR3
      COMMON /EXEC/CASE, TITLE, PAGE, ERROR
      COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1     COMMON/C: /INORM, ISURF, IX, IY, IZ, UI, UL, VI, VL, CR, CT, CHX, N2, N3,
      XO, YO, ZO, API1, API2, API3, AP21, AP22, AP23, AP31, AP32, AP33,
2     FLOWC(2), XI(500), YI(500), B(503,2), XKF(503), AA(2477),
      XB(4), YB(4), ZB(4),
3     NDSET, IABSET, IR, LURG(20), ISK(20), NS, ISFR, IVIS, IFTYP, IFLOW,
4     BX(2,20), BY(2,20), NB, DUM(4)
C
      DIMENSION TITLE(15), E4(12), E4B(12), E1(24), LOSF(5),
      FLOW(503, 2),
      XT(6),
C
      INTEGER ERROR, PAGE, CASE
      INTEGER TAPEIN, TAPEOT, TAPEA, TAPEB, TAPPC, TAPFD, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1
C     EQUIVALENCE (FLOW(1,1), B(1,1))
C
      DATA RC, NX, MX, MX1, KD
      /0, 1745329E-1, 503, 2, 1, 2477/
C     SET CHECKOUT PRINT FLAG
      IPRINT = 1
      IC = 0
      NB = 0
      MM = MX
C
C     INITIALIZE CONSTANT ARRAY FLOW
      DO 98 I = 1, MX
      98 FLOWC(I) = 0.0
C
C
```

SFNTR3

```

C CYCLE ON SUB-REGIONS
DO 600 II = 1, NS
  ISBR = ISR(II)
  IG10 = LOR(IR) + 4 + ISBR
  CALL READMS(10, E1, 25, IG10)
  READ(10, IG10) E1
  LOFF = E1(1) + 0.01
  N1 = E1(6) + 0.01

C
  DO 500 J = 1, 5
    LOSF(J) = E1(J+20) + 0.01
  WRITE(TAPEOT, 2000) II, ISBR, LOFF, N1, LOSF
2000 FORMAT(1H0, 5HII = , I3, 5X7HISBR = , I3, 5X7HLOFF = , I5,
  1 5X5HNI = , I5, 5X7HLOSF = , 5I10)
  IF (ISFR .EQ. 0) GO TO 510

C
  IF (LOSF(ISFR) .LE. 0) GO TO 600

C
C READ IN SECONDARY FLOW POINTERS
  IG10 = LOSF(ISFR)
  CALL READMS(10, E1, 25, IG10)
  READ(10, IG10) E1
  LOFF = E1(1) + 0.01
  N1 = E1(6) + 0.01
  WRITE(TAPEOT, 2010) ISFR, LOFF, N1
2010 FORMAT(1H0, 7HISFR = , I3, 5X7HLOFF = , I5, 5X5HNI = , I5)
C

C READ IN THE DATA
  510 ITAG10 = LOFF
  ISF = 0
  IB = 0
  DO 310 I = 1, NI
    CALL READMS(10, E4, 12, ITAG10)
    READ(10, ITAG10) E4
  WRITE(TAPEOT, 990) ITAG10, E4

```

```

SFN3 036
SFN3 037
SFN3 038
SFN3 039
SFN3 040C
SFN3 041X
SFN3 042
SFN3 043
SFN3 044
SFN3 045
SFN3 046
SFN3 047
SFN3 048
SFN3 049
SFN3 050
SFN3 051
SFN3 052
SFN3 053
SFN3 054
SFN3 055
SFN3 056C
SFN3 057X
SFN3 058
SFN3 059
SFN3 060
SFN3 061
SFN3 062
SFN3 063
SFN3 064
SFN3 065
SFN3 066
SFN3 067
SFN3 068C
SFN3 069X
SFN3 070

```

SFNTR3

```
C 990 FORMAT (1M,7HITAG10=,15,4H EQ,6E12,5,71H ,16X,6E12,5)
      ITAG10 = ITAG10 + 1
C
C TRANSFORM DATA
301 CONTINUE
      XX = EQ(1) = XC
      YY = EQ(2) = YD
      ZZ = EQ(3) = ZO
C
      XT(1) = XX*AP11 + YY*AP12 + ZZ*AP13
      XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
      XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
      XT(4) = XT(1)
      XT(5) = SQRT(XT(2)**2 + XT(3)**2)
      IF (XT(2) .EQ. 0.0) GO TO 304
      XT(6) = ATAN2(XT(2),XT(3))
      GO TO 305
304 XT(6) = 0.0
      IF (XT(3) .GT. 0.0) XT(6) = 3.141592654
305 CONTINUE
      IF (IBSF .EQ. 1) GO TO 523
C
C NORMALIZE THE DATA
      IC = IC + 1
      IF (IC .LE. 500) GO TO 303
      WRITE (TAPEOT,302)
302 FOR-PAT (1M,40H*ROUTINE SFNTRP HAS ATTEMPTED TO LOAD MORE DATA ,
      1 62HPOINTS INTO INTERPOLATION ARRAYS THEN WE HAVE SPACE AVAILABLE,
      2 71H 52H*CALCULATIONS WILL CONTINUE WITH ONLY THE FIRST 500 ,
      3 7HPOINTS. )
      IC = IC - 1
      GO TO 311
303 VI(IC) = (XT(IY) = VI)/VL
      IF (ISUPP .EQ. 1) GO TO 306
      XI(IC) = (XY(IX) = UI)/UL
```

SFN3 071
SFN3 072
SFN3 073
SFN3 074
SFN3 075
SFN3 076
SFN3 077
SFN3 078
SFN3 079
SFN3 080
SFN3 081
SFN3 082
SFN3 083
SFN3 084
SFN3 085
SFN3 086
SFN3 087
SFN3 088
SFN3 089
SFN3 090
SFN3 091
SFN3 092
SFN3 093
SFN3 094
SFN3 095
SFN3 096
SFN3 097
SFN3 098
SFN3 099
SFN3 100
SFN3 101
SFN3 102
SFN3 103
SFN3 104
SFN3 105

SFNTR3

SFNTR3

```

GO TO 307
306 CHX = CR + (CT*CR)*YI(IC)
XLF = UI + (UL*UI)*YI(IC)
XI(IC) = (XT(IX) - XLE)/CHX
C
307 CONTINUE
C CHECK IF WITHIN SURFACE BOUNDARIES.
IF (XI(IC) .LT. 0.0) GO TO 308
IF (XI(IC) .GT. 1.0) GO TO 30A
IF (YI(IC) .LT. 0.0) GO TO 30A
IF (YI(IC) .GT. 1.0) GO TO 30B
C
C POINT WITHIN SURFACE BOUNDARIES, SET UP FLOW BOUNDARIES **
C ** INITIALIZE AS SURFACE BOUNDARIES,
IF (IB .EQ. 1) GO TO 515
IB = 1
NB = NA + 1
BX(1,NB) = 0.0
BY(1,NB) = YI(IC)
IF (ISFR .EQ. 0) GO TO 520
IF ((IVIS .EQ. 1) .AND. (ISFR .EQ. 1)) GO TO 520
RX(1,NB) = XI(IC)
C
520 CONTINUE
IF (IFTYP .EQ. 2) GO TO 530
C CYCLE REMAINING SECONDARY FLOWS TO ESTABLISH BOUNDARIES.
ISF2 = ISFR
521 ISF2 = ISF2 + 1
IF (ISF2 .GT. 5) GO TO 530
IG10SF = LOSF(ISF2)
IF (IG10SF .LE. 0) GO TO 521
CALL READMS(10,E1,25,IG10SF)
READ(10,IG10SF) E1
LOFF = E1(1) + 0.01
CALL READMS(10,E4B,12,LOFF)

```

SFN3 106
 SFN3 107
 SFN3 108
 SFN3 109
 SFN3 110
 SFN3 111
 SFN3 112
 SF 113
 SFN3 114
 SFN3 115
 SFN3 116
 SFN3 117
 SFN3 118
 SFN3 119
 SFN3 120
 SFN3 121
 SFN3 122
 SFN3 123
 SFN3 124
 SFN3 125
 SFN3 126
 SFN3 127
 SFN3 128
 SFN3 129
 SFN3 130
 SFN3 131
 SFN3 132
 SFN3 133
 SFN3 134
 SFN3 135
 SFN3 136
 SFN3 137C
 SFN3 138I
 SFN3 139
 SFN3 140C

SFNTR3

SFNTR3

```

C      READ(10,LOFF)  E4B
      E4(1) = E4B(1)
      E4(2) = E4B(2)
      E4(3) = E4B(3)
      IBSF = 1
      GO TO 301
523 IBSF = 0
      YIB = (XT(IY) - V1)/VL
      IF (ISURF.EG, 1) GO TO 526
      XIR = (XT(IX) - U1)/UL
      GO TO 527
526 CHA = CR + (CT=CR)*YIB
      XLR = U1 + (UL=U1)*YIB
      XIR = (XT(IX) - XLR)/CHB
C      527 CONTINUE
      BX(2,NB) = XIB
      BY(2,NB) = YIB
      GO TO 535
C
C
C      530 CONTINUE
      BX(2,NB) = 1.0
      BY(2,NB) = BY(1,NB)
C      POINT WITHIN BOUNDARIES.  SET UP FLOW ARRAYS.
C      535 CONTINUE
C
      FLOW(IC,1) = ALOG(E4(5))
      FLOW(IC,2) = 0.0
      GO TO 310
C
C      POINT NOT WITHIN BOUNDARIES.  RESET COUNTER AND GO TO NEXT POINT.
C      308 IC = IC + 1
C

```

SFN3 1411
 SFN3 142
 SFN3 143
 SFN3 144
 SFN3 145
 SFN3 146
 SFN3 147
 SFN3 148
 SFN3 149
 SFN3 150
 SFN3 151
 SFN3 152
 SFN3 153
 SFN3 154
 SFN3 155
 SFN3 156
 SFN3 157
 SFN3 158
 SFN3 159
 SFN3 160
 SFN3 161
 SFN3 162
 SFN3 163
 SFN3 164
 SFN3 165
 SFN3 166
 SFN3 167
 SFN3 168
 SFN3 169
 SFN3 170
 SFN3 171
 SFN3 172
 SFN3 173
 SFN3 174
 SFN3 175

SFNTP3

SFNTR3

```

310 CONTINUE
600 CONTINUE
   IF (IC .GT. 2) GO TO 311
   WRITE(TAPEOT,1000) IC
1000  JRMAT(1M1, 5MONLY ,I3,29H POINTS WERE FOUND IN SFNTR3 ,
      1 56HTO BE WITHIN SURFACE BOUNDARIES. THIS IS INSUFFICIENT /
      2 55HTO USE THE SURFACE SPLINE. RUN CONTINUES, BUT USER IS,
      3 34H CAUTIONED TO CHECK RESULTS CAREFULLY.)
      WRITE(TAPEOT,1005) NDSET,IABSET,IR,ISR,ISFR
1005  FORMAT(1M0,40HTHIS APPLIES TO THE FOLLOWING FLOW DATA /
      1 12H DATA SET = ,I3, 10X17HALPHA=BETA SET = ,I3,
      2 10X14FLOW REGION = ,I3/
      3 15H SUB-REGIONS = , 20I3/
      4 14H SECONDARY FLOW = ,I3)
      ERROR = 1
      RETURN
C
C ALL THE SURFACE DATA ARE IN PROPER ARRAYS.
C
311 N2 = IC
C
C CHECK BOUNDARY POINTS
   IF (NB .GT. 1) GO TO 700
   NB = 2
   BX(1,2) = BX(1,1)
   BX(2,2) = BX(2,1)
   BY(1,2) = 1.0
   BY(2,2) = 1.0
C
C NORMALIZE TO FLOW BOUNDARIES (IN X1 ONLY)
C FIRST ARRANGE BOUNDARY DATA IN ORDER OF ASCENDING Y1 VALUES
700 IB = 0
710 IA = IB + 1
   II = 1
720 YY = BY(IB,11)

```

SFN3 176
SFN3 177
SFN3 178
SFN3 179
SFN3 180
SFN3 181
SFN3 182
SFN3 183
SFN3 184
SFN3 185
SFN3 186
SFN3 187
SFN3 188
SFN3 189
SFN3 190
SFN3 191
SFN3 192
SFN3 193
SFN3 194
SFN3 195
SFN3 196
SFN3 197
SFN3 198
SFN3 199
SFN3 200
SFN3 201
SFN3 202
SFN3 203
SFN3 204
SFN3 205
SFN3 206
SFN3 207
SFN3 208
SFN3 209
SFN3 210

SFNTR3

SFNTPS

```

I2 = I1 + 1
J = 0
DO 730 I = I2,NB
  IF (YY .LT. BY(IB,I)) GO TO 730
  J = I
  YY = BY(IB,I)
730 CONTINUE
C
  IF (J .EQ. 0) GO TO 740
  BY(IB,J) = BY(IB,I)
  BY(IB,I) = YY
C
  740 I1 = I1 + 1
  IF (I1 .LT. NB) GO TO 720
  IF (IB .EQ. 1) GO TO 710
C
C BOUNDARY DATA NOW IN ORDER, PROCEED WITH NORMALIZATION.
  IF (IPRINT .EQ. 0) GO TO 770
  WRITE(TAPEOT,750) NB
750 FORMAT(1H1,16HBOUNDARY DATA, , I3, 4H POINTS, /
  1 1HD, 3X1H1, 7X2HX1, 12X2HY1, 12X2HX2, 12X2HY2//)
  WRITE(TAPEOT,760) (I,BX(1,I),BY(1,I),BX(2,I),BY(2,I), I=1,NB)
760 FORMAT(1H , I3, 4F14.6)
770 CONTINUE
  IC = 0
  DO 900 I = 1,N2
    XX = XI(I)
    YY = YI(I)
    IB = 0
  800 IB = IB + 1
C
  DO 810 I1 = 1,NB
    J = I1
    IF (YY .LT. BY(IB,I1)) GO TO 820
  810 CONTINUE

```

SFNS 211
SFNS 212
SFNS 213
SFNS 214
SFNS 215
SFNS 216
SFNS 217
SFNS 218
SFNS 219
SFNS 220
SFNS 221
SFNS 222
SFNS 223
SFNS 224
SFNS 225
SFNS 226
SFNS 227
SFNS 228
SFNS 229
SFNS 230
SFNS 231
SFNS 232
SFNS 233
SFNS 234
SFNS 235
SFNS 236
SFNS 237
SFNS 238
SFNS 239
SFNS 240
SFNS 241
SFNS 242
SFNS 243
SFNS 244
SFNS 245

SFNTPS

```

C      820 IF (J .EQ. 1) J = 2
      DY = BY(IB,J) - BY(IB,J-1)
      IF (DY .EQ. 0.0) GO TO 900
      DX = BX(IB,J) - BX(IB,J-1)
      BP(IB) = BX(IB,J-1) + (YY = BY(IB,J-1))*DX/DY
      GO TO (830,840), IB
      830 IF (XX .LT. BP(IB)) GO TO 900
      GO TO 800
      840 IF (XX .GT. BP(IB)) GO TO 900

C      POINT WITHIN FLOW BOUNDARIES
      IC = IC + 1
      XI(IC) = XX = BP(I)
      YI(IC) = YY
      IF (I VIS .EQ. 0) GO TO 850
      XI(IC) = XI(IC)/(BX(2,1) - BX(1,1))
      GO TO 860
      850 XI(IC) = XI(IC)/(BP(2) - BP(1))
      860 FLOW(IC,1) = FLOW(I,1)
      FLOW(IC,2) = FLOW(I,2)

C      DO 300 II = 1,MM
      300 FLOWC(II) = FLOWC(II) + FLOW(IC,II)

C      900 CONTINUE
      IF (IC .GT. 2) GO TO 910
      WRITE(TAPEOT,1010) IC
      1010 FORMAT(IH1, 5HONLY ,I3, 29H POINTS WERE FOUND IN SFNTR3,
      15HNTO BE WITHIN FLOW BOUNDARIES. THIS IS INSUFFICIENT/
      2 55H TO USE THE SURFACE SPLINE. RUN CONTINUES, BUT USER IS,
      3 38H CAUTIONED TO CHECK RESULTS CAREFULLY.)
      WRITE(TAPEOT,1005) MDSET,IABSET,IR,ISSR,ISSFR
      ERROR = I
      RETURN

```

```

8FN3 246
8FN3 247
8FN3 248
8FN3 249
8FN3 250
8FN3 251
8FN3 252
8FN3 253
8FN3 254
8FN3 255
8FN3 256
8FN3 257
8FN3 258
8FN3 259
8FN3 260
8FN3 261
8FN3 262
8FN3 263
8FN3 264
8FN3 265
8FN3 266
8FN3 267
8FN3 268
8FN3 269
8FN3 270
8FN3 271
8FN3 272
8FN3 273
8FN3 274
8FN3 275
8FN3 276
8FN3 277
8FN3 278
8FN3 279
8FN3 280

```

```

910 CONTINUE
N2 = IC
N3 = N2 + 3
C PRINT NORMALIZED ARRAYS
IF (IPRINT.EQ. 0) GO TO 330
WRITE(TAPEOT,3060) NDSSET, IABSET,IR
3060 FORMAT(14I,34HNORMALIZED SURFACE DATA, NDSSET = ,I2, 5X,
1 9HIABSET = ,I2, 5X, 9HREGION = ,I2/I-0, Y4,IHI, Y13,2HXI,
2 Y27,2HYI, T39,4MLGCF, T55,2HNT //)
WRITE(TAPEOT,3070) (I,XI(I), YI(I), (FLOW(I,J),J=1,MX),I=1,N2)
3070 FORMAT(1I, I3, 4F14.6)
C
C 330 CONTINUE
DO 350 J = 1,MX
FLOWC(J) = FLOWC(J)/N2
DO 340 I = 1,N2
340 FLOW(I,J) = FLOW(I,J) * FLOWC(J)
350 CONTINUE
C
C CALCULATE COEFFICIENT ARRAY (SURFACE SPLINE).
LS = 0
CALL ROWFMI (N2,MM,MI,VI,FI,FL,FK,FX,MY,LS,TAPEB)
CALL SOLVIT(AA,N3,MM,KD,TAPEB,TAPEC,TAPEO,TAPEB,6,NX,MX,NERR)
IF (NERR.NE. 0) WRITE(TAPEOT,220)
220 FORMAT(1MI, 16HMATRIX SINGULAR.)
C
C COEFFICIENTS ARE IN ARRAY B(I,J)
IF (IPRINT.EQ. 0) GO TO 400
WRITE(TAPEOT,3080)
3080 FORMAT(1HI,42HSURFACE INTERPOLATION COEFFICIENTS B(I,J)//)
WRITE(TAPEOT,3090) (I, (B(I,J), J=1,MX),I=1,N3)

```

```

SFN3 281
SFN3 282
SFN3 283
SFN3 284
SFN3 285
SFN3 286
SFN3 287
SFN3 288
SFN3 289
SFN3 290
SFN3 291
SFN3 292
SFN3 293
SFN3 294
SFN3 295
SFN3 296
SFN3 297
SFN3 298
SFN3 299
SFN3 300
SFN3 301
SFN3 302
SFN3 303
SFN3 304
SFN3 305
SFN3 306
SFN3 307
SFN3 308
SFN3 309
SFN3 310
SFN3 311
SFN3 312
SFN3 313
SFN3 314
SFN3 315

```

SPNTRJ

3090 FORMAT(IH, 5, 2F15.5,
WRITE(TAPECT, 3100) FL...
3100 FORMAT(IH0, 5X, 2F15.5,
400 CONTINUE

C
C
C
C

RETURN
END

SPN3 316
SPN3 317
SPN3 318
SPN3 319
SPN3 320
SPN3 321
SPN3 322
SPN3 323
SPN3 324
SPN3 325

SPNTRJ

VALU3

VAL3 001
 VAL3 002
 VAL3 003
 VAL3 004
 VAL3 005
 VAL3 006
 VAL3 007
 VAL3 008
 VAL3 009
 VAL3 010
 VAL3 011
 VAL3 012
 VAL3 013
 VAL3 014
 VAL3 015
 VAL3 016
 VAL3 017
 VAL3 018
 VAL3 019
 VAL3 020
 VAL3 021
 VAL3 022
 VAL3 023
 VAL3 024
 VAL3 025
 VAL3 026
 VAL3 027
 VAL3 028
 VAL3 029
 VAL3 030
 VAL3 031
 VAL3 032
 VAL3 033
 VAL3 034
 VAL3 035

```

SURROUTINE VALU3(N,X,Y,Z,XI,YN,ZN,IC,IT,INT,FMT,EMX,EMY,EM7,POP,
1 TOT)
C
COMMON/C1 /NORM,ISURF,IX,IY,IZ,UI,UJ,VL,CR,CT,CHX,N2,NS,
1 XQ,YO,ZO,AP11,AP12,AP13,AP21,AP22,AP23,AP31,AP32,AP33,
2 FLOWC(2),XI(500),YI(500),B(503,2),KKF(503),AA(2477),
A XR(4), YB(4), ZB(4),
3 NDSFT,I,BSFT,IR,LORG(20),ISR(20),NS,ISFR,IVIS, IFTYP,IFLOW,
4 BX(2,20), BY(2,20), NR ,DUM(4)
C
DIMENSION X(N),Y(N),Z(N),INT(N),EMT(V),EMX(N),EMY(N),EMZ(N),
1 POP(N),TMT(N),VT(6),XT(6),FI(7), BP(2)
C
MM = 7
IF (IVIS .EQ. 1) MM = 2
DO 500 I = 1,N
IF (IC .EQ. 1) GO TO 70
C TRANSFORM COORDS
XX = X(I) * XO
YY = Y(I) * YO
ZZ = Z(I) * ZO
C
XT(1) = XX*AP11 + YY*AP12 + ZZ*AP13
XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
XT(4) = XT(1)
XT(5) = SQRT(XT(2)**2 + XT(3)**2)
IF (XT(2) .EQ. 0.0) GO TO 40
XT(6) = A * XT(2) * XT(3)
GO TO 50
40 XT(6) = 0.0
50 CONTINUE
XT(6) = 3.141592654

```

VALU3

VALUS

```
C C NORMALIZE COORDS
  YN = (XT(IY) - V1)/VL
  IF (ISURF .EQ. 1) GO TO 60
  XN = (XT(IX) - U1)/UL
  GO TO 70
  60 CHY = CR + (CT-CR)*YN
  XLF = U1 + (UL-U1)*YN
  XN = (XT(IX) - XLE)/CHX
C C CHECK IF WITHIN SURFACE BOUNDARIES
  70 IF (XN .LT. 0.0) GO TO 80
  IF (XN .GT. 1.0) GO TO 80
  IF (YN .LT. 0.0) GO TO 80
  IF (YN .GT. 1.0) GO TO 80
C C CHECK IF WITHIN FLOW BOUNDARIES
  IB = 0
  800 IB = IB + 1
  DO 810 I1 = 1, NB
  J = I1
  IF (YN .LT. BY(IB, I1)) GO TO 820
  810 CONTINUE
C  820 IF (J .EQ. 1) J = 2
  DY = BY(IB, J) - BY(IB, J-1)
  IF (DY .EQ. 0.0) GO TO 80
  DX = BX(IB, J) - BX(IB, J-1)
  BP(IB) = BX(IB, J-1) + (YN - BY(IB, J-1))*DX/DY
  GO TO (830, 840), IB
  830 IF (XN .LT. BP(IB)) GO TO 80
  GO TO 800
  840 IF (XN .GT. BP(IB)) GO TO 80
C C POINT WITHIN FLOW BOUNDARIES
```

VAL3 036
VAL3 037
VAL3 038
VAL3 039
VAL3 040
VAL3 041
VAL3 042
VAL3 043
VAL3 044
VAL3 045
VAL3 046
VAL3 047
VAL3 048
VAL3 049
VAL3 050
VAL3 051
VAL3 052
VAL3 053
VAL3 054
VAL3 055
VAL3 056
VAL3 057
VAL3 058
VAL3 059
VAL3 060
VAL3 061
VAL3 062
VAL3 063
VAL3 064
VAL3 065
VAL3 066
VAL3 067
VAL3 068
VAL3 069
VAL3 070

VALUS

VALUS

VAL3 071
VAL3 072
VAL3 073
VAL3 074
VAL3 075
VAL3 076
VAL3 077
VAL3 078
VAL3 079
VAL3 080
VAL3 081
VAL3 082
VAL3 083
VAL3 084
VAL3 085
VAL3 086
VAL3 087
VAL3 088
VAL3 089
VAL3 090
VAL3 091
VAL3 092
VAL3 093
VAL3 094
VAL3 095
VAL3 096
VAL3 097
VAL3 098
VAL3 099
VAL3 100
VAL3 101
VAL3 102
VAL3 103
VAL3 104
VAL3 105

```

      XN = XN + BP(I)
      IF (IVIS .EQ. 0) GO TO 850
      XN = XN / (BX(2,I) + BX(1,I))
      GO TO 860
      850 XN = XN / (BP(2) + BP(1))
      860 CONTINUE
C
C POINT WITHIN BOUNDARIES. CALCULATE VALUES.
      XKF(1) = I.0
      XKF(2) = XN
      XKF(3) = YN
      J3 = 3
C
      DO 250 J = 1,N2
      J3 = J3 + 1
      UIJ = 0.0
      TR1 = 0.0
      TR2 = 0.0
      IF (XN .EQ. XI(J)) GO TO 210
      TR1 = (XN - XI(J))**2
      210 IF (YN .EQ. YI(J)) GO TO 220
      TR2 = (YN - YI(J))**2
      220 RBIJ = TR1 + TR2
      IF (RBIJ .EQ. 0.0) GO TO 230
      UIJ = RBIJ * ALOG(RBIJ)
      230 XKF(J3) = UIJ
      250 CONTINUE
C
C
      DO 300 K = 1,MM
      FI(K) = FLOWC(K)
      DO 290 J = 1,N3
      FI(K) = FI(K) + XKF(J) * B(J,K)
      290 CONTINUE
      300 CONTINUE

```

VALUS

VALUS

VAL3	106
VAL3	107
VAL3	108
VAL3	109
VAL3	110
VAL3	111
VAL3	112
VAL3	113
VAL3	114
VAL3	115
VAL3	116
VAL3	117
VAL3	118
VAL3	119
VAL3	120
VAL3	121
VAL3	122
VAL3	123
VAL3	124
VAL3	125
VAL3	126
VAL3	127
VAL3	128
VAL3	129
VAL3	130
VAL3	131
VAL3	132
VAL3	133
VAL3	134
VAL3	135
VAL3	136
VAL3	137
VAL3	138
VAL3	139
VAL3	140

VALUS

```

C
INT(I) = 1
IF (IVIS .EQ. 0) GO TO 305
EMT(I) = EXP(PI(I))
EMX(I) = FI(2)
EMY(I) = 0.0
EMZ(I) = 0.0
POP(I) = 0.0
TOT(I) = 0.0
GO TO 500
305 CONTINUE
EMT(I) = FI(1)
FT = SQRT(FI(2)**2 + FI(3)**2 + FI(4)**2)
IF (FT .EQ. 0.0) FT = 1.0
FI(2) = FI(2)/FT
FI(3) = FI(3)/FT
FI(4) = FI(4)/FT
EMX(I) = FI(2)/CR
EMY(I) = FI(3)/VL
POP(I) = FI(5)
TOT(I) = FI(6)
ZN = FI(7)

C TRANSFORM XM, YN, ZN COORDS BACK TO X, Y, Z
IF (IVIS .EQ. 0) GO TO 310
XN = XN*(BX(2,1) + BX(1,1))
GO TO 320
310 XN = XN*(BP(2) + BP(1))
320 XN = XN + BP(1)
XT(IY) = YN*VL + VI
XT(I7) = ZN
IF (ISURF .EQ. 1) GO TO 75
XT(IX) = XN*UL + UI
GO TO 76
75 XT(IX) = XN* CHX + UI + (UL-UI)*YN

```

VALUS

VAL3	141
VAL3	142
VAL3	143
VAL3	144
VAL3	145
VAL3	146
VAL3	147
VAL3	148
VAL3	149
VAL3	150
VAL3	151
VAL3	152
VAL3	153
VAL3	154
VAL3	155
VAL3	156
VAL3	157
VAL3	158
VAL3	159
VAL3	160
VAL3	161
VAL3	162
VAL3	163
VAL3	164
VAL3	165
VAL3	166
VAL3	167
VAL3	168
VAL3	169
VAL3	170
VAL3	171
VAL3	172
VAL3	173
VAL3	174
VAL3	175

```

76 CONTINUE
  IF (INORM .LT. 0) GO TO 79
  GO TO (81, 82, 83, 84), INORM

C   INORM = 0      IX = 4, IY = 5, IZ = 6
79 XX = XT(IX)
   SINPH = SIN(XT(IZ))
   COSPH = COS(XT(IZ))
   YY = XT(IY)*SINPH
   ZZ = -XT(IY)*COSPH
   VT(1) = FI(2)
   VT(2) = FI(3)*SINPH + FI(4)*COSPH
   VT(3) = FI(4)*SINPH - FI(3)*COSPH
   GO TO 90

C   INORM = 1      IX = 1, IY = 2, IZ = 3
81 XX = XT(IX)
   YY = XT(IY)
   ZZ = XT(IZ)
   110 VT(1) = FI(2)
       VT(2) = FI(3)
       VT(3) = FI(4)
   GO TO 90

C   INORM = 2      IX = 1, IY = 3, IZ = 4
82 XX = XT(IX)
   YY = XT(IZ)
   ZZ = XT(IY)
   111 VT(1) = FI(2)
       VT(2) = FI(4)
       VT(3) = FI(3)
   GO TO 90

C   INORM = 3      IX = 2, IY = 3, IZ = 1
83 XX = XT(IZ)

```

VALUS

VALUS

VAL3 176
VAL3 177
VAL3 178
VAL3 179
VAL3 180
VAL3 181
VAL3 182
VAL3 183
VAL3 184
VAL3 185
VAL3 186
VAL3 187
VAL3 188
VAL3 189
VAL3 190
VAL3 191
VAL3 192
VAL3 193
VAL3 194
VAL3 195
VAL3 196
VAL3 197
VAL3 198
VAL3 199
VAL3 200
VAL3 201
VAL3 202
VAL3 203
VAL3 204
VAL3 205
VAL3 206
VAL3 207
VAL3 208
VAL3 209
VAL3 210

```

      VY = XT(IY)
      ZZ = YT(IY)
112  VT(1) = FI(4)
      VT(2) = FI(2)
      VT(3) = FI(3)
      GO TO 90

C      INORM = 4, IY = 4, IY = 6, IZ = 5
04  XX = XT(IY)
      SINPH = SIN(XT(IY))
      COSPH = COS(XT(IY))
      VY = XT(IZ)*SINPH
      ZZ = XT(IZ)*COSPH
      EMY(I) = EMY(I)/ZN
113  VT(1) = FI(2)
      VT(2) = FI(4)*SINPH + FI(3)*COSPH
      VT(3) = -FI(4)*COSPH + FI(3)*SINPH

C      90  IF (IC .EQ. 6) GO TO 100
      X(I) = XG + XX*AP11 + YY*AP21 + ZZ*AP31
      Y(I) = YG + XX*AP12 + YY*AP22 + ZZ*AP32
      Z(I) = ZG + XX*AP13 + YY*AP23 + ZZ*AP33

C      CHECK COORDINATE SYSTEM FOR DIRECTION COSTNES
100  IF (IT .EQ. 1) GO TO 500
120  EMY(I) = VT(1)*AP11 + VT(2)*AP21 + VT(3)*AP31
      EMY(I) = VT(1)*AP12 + VT(2)*AP22 + VT(3)*AP32
      EMY(I) = VT(1)*AP13 + VT(2)*AP23 + VT(3)*AP33
      GO TO 500

C      PRINT NOT WITHIN BOUNDARIES. SET FLAG AND GO TO NEXT POINT.
05  INT(I) = 0

C
C      500 CONTINUE

```

VALUS

VALUS

VALS 211
VALS 212
VALS 213
VALS 214

C
C

RETURN
END

VALUS

```

C
C
C
C
SUBROUTINE LANNAR
      THIS ROUTINE WAS TAKEN FROM NASA TN D-5681 BY W.D. McNALLY.

COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH, ALT, PSTAG, STAG, V, RENC, PFS, TFS, AFS, RHDFS, VIS
COMMON /TAPE/TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON /GASD/GAM, GASCP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
INTEGER TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
COMMON/CI/ R, PTZ, TZZ, UPWACH, NST, NVP, NTURB, KPVM, XEM, KSMTH,
      KSPLN, KLE, KATCH, CTHT, DLAM, ILAM, DTURB, TTURB, KPFE, KGRAD, KSDE, KLAM,
      2KMAIN, KPROF, X(100), Y(100), PRES(100), UE(100), ME(100), POPTZ(100),
      3VOYCP(100), TVAL(100),
      4 PSZ, TSZ, UZ, ASZ, ATZ, PHSZ, RMTZ, MUSZ, MUTZ, NUSZ, NUTZ, CP,
      5PR, TC, ARCL,
      7 S(100), SOL(100), AE(100), TSE(100),
      8YAWL(100), YAWT(100), TBAR(100), RM(300), SW(100), SUTHL(100),
      9RMSW(100), RMSE(100), HEADW(100), HEADE(100), NUM(100), MUBAR(100),
      AAAA(100), BB(100), FF(100), DUDS(100), DMDs(100), DMDL(100),
      B THET(100), DELSR(100), DELTA(100), FORM(100),
      CFORMT(100), FORMTR(100), RTH(100), RTHI(100), CR(100),
      DTAWW(100), NUSS(100), DTDY(100), HTRAN(100), CRN(100),
      E SHAPL(100), SHAPK(100), B, NS,
      F FTRAN, FORMS,
      G INST, ITRAN, ISEP,
      H XTAB(103), YTAB1(103), YTAB2(103), NTAB,
      I EROR, TRANS, SEPRN
      DIMENSION COPLN(100), COHML(100), SHEAR(100), DTM(100)
      DIMENSION CCM(20), CCHR(20), CDIF(20), CSHR(20), CCRN(20), CDTH(20)
      DIMENSION STAR(103), CTAB1(103), CTAB2(103)
      DIMENSION CSHRE(16), CCRNE(16), COTWE(16)
      PFAL, MUSZ, NUSZ, MUTZ, NUTZ, MF, NUM, MUBAR, NUSS, NURW, KBAR, INT1, INT2

```

```

LANN 001
LANN 002
LANN 003
LANN 004
LANN 005
LANN 006
LANN 007
LANN 008
LANN 009
LANN 010
LANN 011
LANN 012
LANN 013
LANN 014
LANN 015
LANN 016
LANN 017
LANN 018
LANN 019
LANN 020
LANN 021
LANN 022
LANN 023
LANN 024
LANN 025
LANN 026
LANN 027
LANN 028
LANN 029
LANN 030
LANN 031
LANN 032
LANN 033
LANN 034
LANN 035

```

LAMNAR

LAMN 036
LAMN 037
LAMN 038
LAMN 039
LAMN 040
LAMN 041
LAMN 042
LAMN 043
LAMN 044
LAMN 045
LAMN 046
LAMN 047
LAMN 048
LAMN 049
LAMN 050
LAMN 051
LAMN 052
LAMN 053
LAMN 054
LAMN 055
LAMN 056
LAMN 057
LAMN 058
LAMN 059
LAMN 060
LAMI 061
LAMN 062
LAMN 063
LAMN 064
LAMN 065
LAMN 066
LAMN 067
LAMN 068
LAMN 069
LAMN 070

LAMNAR

```

C
LOGICAL EROR, TRANS, SEPRN
EXTERNAL FUNCT, INT1, INT2

EQUIVALENCE (CSHRE(1), CSHR(1))
EQUIVALENCE (CCRNE(1), CCRN(1))
EQUIVALENCE (CDTHE(1), CDTH(1))

C
C READ DATA FOR CURLN(1), RCRT, DIFF, SHEAR, CRN, AND DTH CURVE FITS
DATA CCN(1), CCN(2), CCN(3), CCN(4), CCN(5), CCN(6) /
1 -.08178, .0667, .03143, .00873, .01657, .01052 /
DATA CRCR(1), CRCR(2), CRCR(3), CRCR(4), CRCR(5), CRCR(6) /
1 5.47073, 43.6053, 227.198, 2067.04, 27372.7, 13691.2 /
DATA CDIF(1), CDIF(2), CDIF(3), CDIF(4), CDIF(5), CDIF(6) /
1 903.785, 26365.0, 3.85695E+5, 1.11044E+6, 4.53853E+7, 7.70276E+7 /
DATA CSHRF / .224488, .1, .21539, .9, .899, .68, .13488,
1 .0, .12, .1, 4768, .10, 52925, .152, 2781, .0, 2406, .0, 015629,
1 .1, 45743, .126, 23395, .000752, .005385, .9, 7438, .39, 40644 /
DATA CCRNE / 2.02056, .19, 7211, .24, 0495, .1400, 002,
1 .050979, .10, 88012, 62, 4419, .5081, 76, .014303, 2.279843,
1 129, 7008, .6257, 848, .0270567, .1, 077051, .57, 4397, .2552, 266 /
DATA CDTHE / 8.02829, .0, 30978, 86, 6244, 36, 4336,
1 2, 71101, .7, 43259, 242, 293, .16, 293, .1, 16393, .07, 61942, 286, 9795,
1 64, 11186, .1, 16758, .3, 70289, 136, 8107, 111, 3276 /
FUNCT(XI, X1, X2, Y1, Y2) = (X1-XI)*(Y2-Y1), (X2-XI) + Y1

C
C INITIALIZE PARAMETERS
INST = 0
ITRAN = 0
ISFP = 0
CF(1) = 0.
TAUW(1) = 0.
MUSS(1) = 0.
DTPY(1) = 0.
HTRAN(1) = 0.

```

LAMNAR

LAMN 071
LAMN 072
LAMN 073
LAMN 074
LAMN 075
LAMN 076
LAMN 077
LAMN 078
LAMN 079
LAMN 080
LAMN 081
LAMN 082
LAMN 083
LAMN 084
LAMN 085
LAMN 086
LAMN 087
LAMN 088
LAMN 089
LAMN 090
LAMN 091
LAMN 092
LAMN 093
LAMN 094
LAMN 095
LAMN 096
LAMN 097
LAMN 098
LAMN 099
LAMN 100
LAMN 101
LAMN 102
LAMN 103
LAMN 104
LAMN 105

```
CRN(1) = 0.  
RTRANE 0.  
  
C CHECK CONSISTENCY OF INITIAL VALUES  
C  
C  
  IF (DLAM,GE.0.,AND,TLAM,GE.0.,AND,DTURB,GE.0.,AND,TTURB,GE.0.)  
  10 GO TO 10  
  ERROR = .TRUE.  
  WRITE (TAPE07,1000)  
  RETURN  
  IF (NTURB,NE.1) GO TO 30  
  ITRAN = 1  
  IF (DTURB,GT.0.,AND,TYURB,GT.0.) GO TO 20  
  FROR = .TRUE.  
  WRITE (TAPE07,1010)  
  RETURN  
  20 IF (UE(1),GT.0.) GO TO 240  
  ERROR = .TRUE.  
  WRITE (TAPE07,1020)  
  RETURN  
  
C BEGIN CALCULATION IN LAMINAR REGION - CHECK FOR INITIAL VALUES  
C CALCULATE INITIAL CORRELATION NUMBER  
C  
  30 IF (DLAM,EG.0.,AND,TLAM,EG.0.) GO TO 70  
  IF (UE(1),GT.0.) GO TO 40  
  ERROR = .TRUE.  
  WRITE (TAPE07,1030)  
  RETURN  
  40 IF (TLAM,EG.0.) GO TO 50  
  INITIAL MOMENTUM THICKNESS WAS GIVEN  
  YEM1 = 1. + S*(GAM-1.)**ME(1)**2  
  CORML(1) = CATZ*TLAM**2/(NUTZ*SUTHL(1)*AFGL)*PRES(1)/PTZ  
  1 *TEM1**1.5  
  CORLN(1) = CORML(1)*DMDL(1)
```

LAMNAR

LAMNAR

```

C          GO TO 90
C INITIAL DISPLACEMENT (THICKNESS WAS GIVEN
50 IF (ABS(DMDL(1)).GE..0001) GO TO 60
CORLN(1)= 0.
TEM1= 1.+5*(GAM01.)*ME(1)**2
FORM(1)= 2.38411*(1.+2.7901.78*PR**.5)*((1.+SW(1))*TEM101.))+(4
15*PR**(1./3.)*3.65*PR*.5)*PR**.5*(TEM101.)
THET(1)= DLAM/FORM(1)
CORML(1) = ATZ*THET(1)**2/(NUTZ*SUTHL(1)*ARCL)*PRES(1)/PTZ
1      *TEM1**1.5
GO TO 90
60 IF (DMDL(1).GT.0.) CALL ROOT(01,0.,DLAM,FUNCT.,SE=5,CORLN(1),SL)
IF (DMDL(1).LT.0.) CALL ROOT( 0.,2.,DLAM,FUNCT.,SE=5,CORLN(1),SL)
CORML(1) = CORLN(1)/DMDL(1)
GO TO 90
C NO INITIAL LAMINAR VALUES GIVEN
C CALCULATE INITIAL CORRELATION NUMBER
C SHARP LEADING EDGE
70 IF (KLE.NE.1.AND.ABS(DMDL(1)).GE..0001) GO TO 80
CORLN(1)= 0.
CORML(1)= 0.
GO TO 90
C STAGNATION POINT
80 CALL CURVFT(CCN,CORLN(1),SW(1),0,5,0)
CORML(1)= CORLN(1)/DMDL(1)
IF (CORML(1).LT.0.) GO TO 90
ERROR = TRUE.
WRITE (TAPE07,1040)
RETURN
C SOLVE LAMINAR DIFFERENTIAL EQUATION
C CALCULATE CORRELATION NUMBERS ALONG THE SURFACE
C

```

LAMN 106
LAMN 107
LAMN 108
LAMN 109
LAMN 110
LAMN 111
LAMN 112
LAMN 113
LAMN 114
LAMN 115
LAMN 116
LAMN 117
LAMN 118
LAMN 119
LAMN 120
LAMN 121
LAMN 122
LAMN 123
LAMN 124
LAMN 125
LAMN 126
LAMN 127
LAMN 128
LAMN 129
LAMN 130
LAMN 131
LAMN 132
LAMN 133
LAMN 134
LAMN 135
LAMN 136
LAMN 137
LAMN 138
LAMN 139
LAMN 140

LAMNAR

```

90 TEM1 = 1. + 5*(GAM*1.)*ME(1)**2
   TEM2 = (3.*GAM*1.)/(2.*GAM*2.)
   DEL = 0.002 * ARCL
   NSTEP = 0
   SS = -DEL
   NTAB = 1
   CTAB1(1) = CORLN(1)
   CTAB2(1) = CORML(1)
   STAB(1) = 0.
   CTABIN = CTARI(1)
   CTAB2N = CTAR2(1)
   IF (KSD E, ME, 1) GO TO 100
   WRITE(TAPEOT,1070) NSTEP,STAB(1),CTABIN,SW(1),ME(1),
1   DMDL(1),CF(1),TEM1
100 SS = SS+DEL
   SSDEL = SS+DEL
   IF (SS .LT. S(2)) GO TO 101
   CALL LGRNGE(S,SW,NST,RS,ANS1)
   CALL LGRNGE(S,ME,NST,SS,ANS2)
   CALL LGRNGE(S,PRES,NST,SS,ANSPI)
   GO TO 102
101 ANS1 = FUNCY (SS,S(1),S(2),SW(1),SW(2))
   ANS2 = FUNCY (SS,S(1),S(2),ME(1),ME(2))
   ANSP1 = FUNCY(SS,S(1),S(2),PRES(1),PRES(2))
102 IF (SSDEL .LT. S(2)) GO TO 103
   CALL LGRNGE(S,ME,NST,SSDEL,ANS3)
   CALL LGRNGE(S,DMDL,NST,SSDEL,ANS4)
   CALL LGRNGE(S,PRES,NST,SSDEL,ANS2)
   GO TO 104
103 ANS3 = FUNCY (SSDEL,S(1),S(2),ME(1),ME(2))
   ANS4 = FUNCY (SSDEL,S(1),S(2),DMDL(1),DMDL(2))
   ANSP2 = FUNCY(SSDEL,S(1),S(2),PRES(1),PRES(2))
104 CONTINUE
   A1 = 0.43631-0.00367*ANS1+0.00481*ANS1**2+0.00651*ANS1**3

```

```

LANN 141
LANN 142
LANN 143
LANN 144
LANN 145
LANN 146
LANN 147
LANN 148
LANN 149
LANN 150
LANN 151
LANN 152
LANN 153
LANN 154
LANN 155
LANN 156
LANN 157
LANN 158
LANN 159
LANN 160
LANN 161
LANN 162
LANN 163
LANN 164
LANN 165
LANN 166
LANN 167
LANN 168
LANN 169
LANN 170
LANN 171
LANN 172
LANN 173
LANN 174
LANN 175

```

LAMNAR

```

A2= 5.43220+2.25400**ANS1-0.06672**ANS1**2=0.20637**ANS1**3
A3= 4.51903-10.49775**ANS1-12.71732**ANS1**2=2.05270**ANS1**3
A4= 19.01831+62.76597**ANS1+115.00986**ANS1**2+62.53113**ANS1**3
A = A1 = A3*CTABIN**2 = 2.**A4*CTABIN**3
B = A2 + 2.**A3*CTABIN + 3.**A4*CTABIN**2
KI= 0
SOL1 = SS/ARCL
SOL2 = SDEL/ARCL
TEM3 = SIMPS1(SOL1,SOL2,INT1,K1)
IF (TEM3.EQ.0..OR.K1.EQ.0) GO TO 110
ERROR = .TRUE.
WRITE (TAPE0Y,1050)
RETURN
110 IF (SS .GT. 0.0) TEM4 = ANS2**(-8)*SQRT(TEM1)*PTZ/ANSPI
    TEM1= 1.+5*(GAM=1.)*ANS3**2
    TEM5 = ANS3**(-8)*SQRT(TEM1)*PTZ/ANSP2
    TEM6 = -A*TEM5*TEM3
    TEM7 = 0.0
    IF (SS .GT. 0.0) TEM7 = TEM5/TEM4*CTAR2N
    CTAB2N = TEM6 + TEM7
    CTABIN = CTAR2N**ANS4
    NSTEP = NSTEP + 1
    IF (K5DE .EQ. 1) WRITE(TAPE0Y,1070) NSTEP,SDEL,CTABIN,
1   ANS1,ANS2,ANS3,ANS4,B,TEM1,TEM3,TEM4,TEM5,TEM6,TEM7
    IF (CTABIN .GT. =0.32) GO TO 122
    WRITE(TAPE0Y,121) NSTEP,SDEL,CTABIN
121 FORMAT(1H0,51H**CTAB1 IS BEING USED BELOW THE RANGE OF THE CURVE ,
2 21H CTAB1 IS SET = *0.32 )
    CTABIN = *0.32
122 CONTINUE
    IF (NSTEP .NE. 5) GO TO 100
    NSTEP = 0
    NTAB = NTAB+1
    CTAB2(NTAB) = CTAB2N

```

LAMN 176
LAMN 177
LAMN 178
LAMN 179
LAMN 180
LAMN 181
LAMN 182
LAMN 183
LAMN 184
LAMN 185
LAMN 186
LAMN 187
LAMN 188
LAMN 189
LAMN 190
LAMN 191
LAMN 192
LAMN 193
LAMN 194
LAMN 195
LAMN 196
LAMN 197
LAMN 198
LAMN 199
LAMN 200
LAMN 201
LAMN 202
LAMN 203
LAMN 204
LAMN 205
LAMN 206
LAMN 207
LAMN 208
LAMN 209
LAMN 210

LAMNAR

LAMNAR

LAMN 211
LAMN 212
LAMN 213
LAMN 214
LAMN 215
LAMN 216
LAMN 217
LAMN 218
LAMN 219
LAMN 220
LAMN 221
LAMN 222
LAMN 223
LAMN 224
LAMN 225
LAMN 226
LAMN 227
LAMN 228
LAMN 229
LAMN 230
LAMN 231
LAMN 232
LAMN 233
LAMN 234
LAMN 235
LAMN 236
LAMN 237
LAMN 238
LAMN 239
LAMN 240
LAMN 241
LAMN 242
LAMN 243
LAMN 244
LAMN 245

LAMNAR

```

CTABI(NTAB)= CTAB2N*ANS4
STAB(NTAB)= SDEL
IF (CTABI(NTAB) .GT. 0.16) GO TO 120
IF (.8S.LT.ARCL) GO TO 100
120 CONTINUE
C CALCULATE LAMINAR BOUNDARY LAYER PARAMETERS AT EACH STATION
C
130 IF (KLAM,NE.1) GO TO 140
WRITE (TAPEOT,1080)
140 IB= 0
150 IB= I+1
IF (I.EQ.NTURD) ITRANZ=1
IF (S(I).LE.STAB(NTAB)) GO TO 160
IF (CYABI(NTAB) .GT. 0.16) GO TO 155
ERROR=.TRUE.
WRITE (TAPEOT,1090)
RETURN
155 WRITE (TAPEOT,156) NTAB,CTABI(NTAB)
156 FORMAT (1H0,41MCORRELATION NUMBER (CTAB1) IS .67, 0.16,
1 34H LAMINAR SEPARATION HAS OCCURRED.,3X, 5HNTAB=,I3,
2 3X, 6HCTAB1=,F10.5)
160 CALL LGRNGE(STAB,CTABI,NTAB,S(I),CORLN(I))
CALL LGRNGE(STAB,CTAB2,NTAB,S(I),CORML(I))
C OBTAIN SHEAR, GRN, AND DTH FROM CURVE FITS VS CORLN AND SW
CALL CURVFT(CSHR,SHEAR(I),CORLN(I),SW(I),3,3)
CALL CURVFT(CCRN,CRN(I),CORLN(I),SW(I),3,3)
CALL CURVFT(CDTH,DTH(I),CORLN(I),SW(I),3,3)
C CALCULATE OTHER LAMINAR BOUNDARY LAYER PARAMETERS
TEM1= 1.0 + 0.5*(GAM=1.0)*ME(I)*ME(I)
THE7(I)= SQRT(-CORML(I)*NUZT*SUTHL(I)*ARCL*PTZ/
1 (ATZ*PPES(I))*TEH1**1.5))
FORM(I)= (-1.1138*CORLN(I)+2.3441)*(-1.+(2.79+1.78*PR**5))*((1.+
1SW(I))*TEH1+1.))+4.65*PR**1./3.)*3.65*PR**5)*PR**5*(TEM1+1.)
DELSR(I)= THFT(I)*FORM(I)

```

LAMNAR

```

RTH(I)=UE(I)*THET(I)/NUM(I)
FORMI(I)=(FORM(I)*SQRT(PR)*TEM1-1.)/((1.+SW(I))*TEM1)
FORMTR(I)=FORMI(I)*(1.+SW(I))
DELTA(I)=THET(I)*(DTH(I)+(TEM1-1.)*(FORMTR(I)+1.))
SHAPL(I)=DELTA(I)**2/NUM(I)*DUDS(I)
IF (1.EQ.1) GO TO 180
CFRW=2.*SHEAR(I)*SQRT(-SD(I)/ME(I)/CORML(I))
CF(I)=CFRW/SQRT(RW(I))
TAUM(I)=CF(I)*HEADW(I)
NURW=CYRW*PR**3/CRN(I)
NUSS(I)=NURW*SQRT(RW(I))
DUDY(I)=NUSS(I)*(TAWL(I)-TVAL(I))/S(I)
HTRAN(I)=TC*DUDY(I)
IF (S(I).GT.STAB(NYAR) .AND. CTAB1(NYAR).GT.0.16) GO TO 161
IF (TAUM(I).GT.0.) GO TO 180
161 IF (KATCH.NE.0) GO TO 170
ISEP=I
SEPRN=.TRUE.
RETURN
170 ITRAN=2
GO TO 270
180 IF (1.EQ.1 .AND. ABS(UE(I)).LE..0000001) GO TO 190
SHAPK(I)=NUTZ*RTH(I)**2*SUTHL(I)**2*(1.+SW(I))**4/ATZ/ME(I)**2/
1FF(I)/ARCL*DMCL(I)*TFMI**2/(GAM-1.)
GO TO 200
190 SHAPK(I)=0.07
200 RTH(I)=RTH(I)*SUTHL(I)*(1.+SW(I))**2/FF(I)/SQRT(TEM1)
IF (RTH(I).GT.1.0E8) WRITE (TAPECT,202) I,RTH(I),SUTHL(I),
1 FF(I),TEM1,UE(I),THET(I),NUM(I),ANSS
202 FORMAT (1H0,17HRTHI IS TOO LARGE,3X,2HTZ,13,3X,4HRTH, F10.5,3X,
1 6MSUTHL, F10.5,3X,3HFF, F10.5,3X,5HTEM1, F10.5, /1H ,20X,
2 9HUE, F10.5,3X,5HTHETS, F10.5,3X,4HNUW, F10.5,3X,5HANS3, F10.5)
C CALCULATE RCRTY TO CHECK FOR INSTABILITY AND TRANSITION
C
C

```

LAMN 246
LAMN 247
LAMN 248
LAMN 249
LAMN 250
LAMN 251
LAMN 252
LAMN 253
LAMN 254
LAMN 255
LAMN 256
LAMN 257
LAMN 258
LAMN 259
LAMN 260
LAMN 261
LAMN 262
LAMN 263
LAMN 264
LAMN 265
LAMN 266
LAMN 267
LAMN 268
LAMN 269
LAMN 270
LAMN 271
LAMN 272
LAMN 273
LAMN 274
LAMN 275
LAMN 276
LAMN 277
LAMN 278
LAMN 279
LAMN 280

LAMNAR

LAMNAR

```
IF (SHAPK(I) .GT. 0.07) SHAPK(I) = 0.07
CALL CURVFT(CRCR,RCRIT,SHAPK(I),0.5,0)
RCRIT= EXP(RCRIT)
IF(INST,NE,0) GO TO 210

C CHECK FOR INSTABILITY
C
C
      IF(RTHI(I),LT,RCRIT) GO TO 270
      RINS= RTHI(I)
      INST= I
      GO TO 270

C CHECK FOR TRANSITION
C
C
      210 K1= 0
          NS= I
          TEM= SIMP1(SOL(INST),SOL(I),INT2,K1)
          IF (TEM,EG,0.,OR,K1,FG,0) GO TO 220
          FROM= .TRUE.
          WRITE (TAPECT,1100)
          RETURN
      220 KBAR= TEM/(SOL(I)-SOL(INST))
          CALL CURVFT(CDIF,DIFF,KBAR,0.5,0)
          RTRANS RINS+DIFF
          IF(RTHI(I),LT,RTRAN) GO TO 270
          IF (I,LT,NTURB) GO TO 270
          ITRANS =I
          GO TO 270
      230 ITRANS I

C COMPUTE INITIAL VALUE FOR TURBULENT SOLUTION
C
C
      240 TRANS= .TRUE.
          IF (OTURB,EG,0.,AND,TTURB,FG,0.) GO TO 260
          IF (OTURB,GT,0.,AND,TTURB,GT,0.) GO TO 250
```

LAMNAR

LAMNAR

```
ERRR .TRUE.  
WRITE (TAPEOT,1110)  
RETURN  
250 THET(IYTRAN)= TTURB  
FORM(IYTRAN)= DTURB/TTURB  
TEMP= 1.+5*(GAM=1.)*ME(IYTRAN)**2  
FORMI(IYTRAN)= PR*(1./3.)*(TEMP+1.)))/((1.+SW(IYTRAN))  
1*TEMP)  
260 IF (CTHET.GT.0..AND.DTURB.EQ.0..AND.TTURB.EQ.0.) THET(IYTRAN)=  
1CTHET*THET(IYTRAN)  
THPTTR = THET(IYTRAN)*SORT(TTZ/TSE(IYTRAN))*PRES(IYTRAN)/PTZ  
FTRAN= (ME(IYTRAN)*ATZ*THETTR/NUTZ)**1./68  
IF (RTRAN.LE.0.) RTRAN=100.  
FORMSB FORMI(IYTRAN)*0.59329*0.06591*ALOG(RTRAN)*0.001272*(ALOG(RTR  
IAN))**2  
C WRITE (6,9000) FTRAN,FORMS,ME(IYTRAN),ATZ,THETTR,NUTZ,THET(IYTRAN),  
C 1 TSE(IYTRAN),TTZ  
C9000 FORMAT (140,49IN LAMNAR = FTRAN, FORMS, ME(IYTRAN), ATZ, THETTR,  
C 134H NUTZ, THET(IYTRAN), TSE(IYTRAN),TTZ, /1H ,5F20.6, /1H ,4F20.6)  
C IF (DTURB.GT.0..AND.TTURB.GT.0.) FORMS=FORMI(IYTRAN)  
IF (FORMS.LT.176.0) GO TO 265  
WRITE (TAPEOT,261)  
261 FORMAT (140,47IN LAMNAR, FORMS .GE. 176.0 WILL CAUSE TROUBLE ,  
1 50HIN RUNKUT WHEN CALLED BY TURBLN. NET FORMS=176.0. )  
FORMS = 176.0  
265 CONTINUE  
RETURN  
C PRINT OUTPUT  
C  
C 270 IF (KLAM.NE.1) GO TO 280  
IF (INST.EQ.0 .OR. INST.EQ.1) WRITE (TAPEOT,1120) I,CORLN(I),  
1 SHEAR(I),DTH(I),FORMTR(I),SHAPL(I),RTHI(I),SHAPK(I),RCRIT  
IF (INST.NE.0 .AND. INST.NE.1) WRITE (TAPEOT,1130) I,CORLN(I),  
1 SHEAR(I),DTH(I),FORMTR(I),SHAPL(I),RTHI(I),KBAR,DIFF,RTRAN
```

LAMN 316
LAMN 317
LAMN 318
LAMN 319
LAMN 320
LAMN 321
LAMN 322
LAMN 323
LAMN 324
LAMN 325
LAMN 326
LAMN 327
LAMN 328
LAMN 329
LAMN 330
LAMN 331
LAMN 332
LAMN 333
LAMN 334
LAMN 335
LAMN 336
LAMN 337
LAMN 338
LAMN 339
LAMN 340
LAMN 341
LAMN 342
LAMN 343
LAMN 344
LAMN 345
LAMN 346
LAMN 347
LAMN 348
LAMN 349
LAMN 350

LAMNAR

LAMNAR

```
IF (ITRAN.EQ.02) WRITE (TAPE1,1140)
280 IF (ITRAN.EQ.01.OR.ITRAN.EQ.02) GO TO 230
IF (I.EQ.NST) RETURN
GO TO 150

C   FORMAT STATEMENTS
C
1000 FORMAT(////,10X,60H NEGATIVE INITIAL VALUE HAS BEEN GIVEN. THIS
      (IS NOT ALLOWED)
1010 FORMAT(////,10X,75H INITIAL VALUES WERE NOT GIVEN FOR THE TURBULEN
      T BOUNDARY LAYER AT STATION 1)
1020 FORMAT(////,10X,80H INITIAL VALUES WERE GIVEN FOR THE TURBULENT BO
      UNDRARY LAYER AT A STAGNATION POINT)
1030 FORMAT(////,10X,94H INITIAL VALUES OTHER THAN ZERO WERE GIVEN FOR
      THE LAMINAR BOUNDARY LAYER AT A STAGNATION POINT)
1040 FORMAT(////,10X,106H FOR THIS INPUT DATA STATION 1 IS ASSUMED TO B
      E A STAGNATION POINT, SINCE NO INITIAL THICKNESSES ARE GIVEN./
210X,118H IN THIS CASE PRESSURE SHOULD DECREASE INITIALLY. EITHER G
310X AN INITIAL VALUE FOR DISPLACEMENT OR MOMENTUM THICKNESS./
410X,60H OR BEGIN WITH A SHORT REGION OF FAVORABLE PRESSURE GRADIENT
5.)
1050 FORMAT(////,10X,37H ERROR IN COMPUTING INTEGRAL FOR CORLM)
1060 FORMAT (1H1//7X,45H LAMINAR DIFFERENTIAL EQUATION SOLUTION FOR ,
      1 SHCORLN,/,1H0,9H NTA STAB,3X,5H CTAB!,5X,4H ANS1,3X,4H ANS2,5X,
      2 4H ANS3,5X,4H ANS4,3X,4H B ,5X,4H TEM1,5X,4H TEM3,7X,4H TEM4,7X,
      3 4H TEM5,5X,4H TEM6,5X,4H TEM7)
1070 FORMAT (1H ,73,F7.4,F8.4,F7.3,3F5.4,2F7.4,F11.6,2F11.4,2F9.4)
1080 FORMAT(1H1//1X,59H LAMINAR CALCULATION OF INSTABILITY AND TRANSITI
      ON LOCATIONS//1X,7H STATION,2X,5H COR(N,5X,5H SHEAR,5X,3H DTH,6X,6H FO
      2RMTR,4X,5H8MAPL,9X,4H RTHI,4X,5H SHAPK,9.,5H RCRTI,6X,4H KBAR,10X,4H DI
      3FF,9X,SHRTRAN)
1090 FORMAT(////,10X,65H LAMINAR SOLUTION HAS PROCEEDED BEYOND THE RANG
      E WHERE IT IS VALID)
1100 FORMAT(////,10X,36H ERROR IN COMPUTING INTEGRAL FOR KBAR)
1110 FORMAT(////,10X,64H IF INITIAL TURBULENT VALUES ARE GIVEN, THEY BO
```

LAMN 351
LAMN 352
LAMN 353
LAMN 354
LAMN 355
LAMN 356
LAMN 357
LAMN 358
LAMN 359
LAMN 360
LAMN 361
LAMN 362
LAMN 363
LAMN 364
LAMN 365
LAMN 366
LAMN 367
LAMN 368
LAMN 369
LAMN 370
LAMN 371
LAMN 372
LAMN 373
LAMN 374
LAMN 375
LAMN 376
LAMN 377
LAMN 378
LAMN 379
LAMN 380
LAMN 381
LAMN 382
LAMN 383
LAMN 384
LAMN 385

LAMNAR

LAMNAR

```
1120 FORMAT(1A,1X,3F10.4,1X,F12.1,1X,F10.5,1X,F12.1)
1130 FORMAT(1A,1X,5F10.4,1X,F12.1,2D12.5,1X,F12.1,1X,F12.1)
1140 FORMAT(////,10X,85HLAMINAR SEPARATION HAS OCCURRED. ASSUMED TO BE
      1 TRANSITION TO TURBULENT BOUNDARY LAYER)
      END
```

LAMN 386
LAMN 387
LAMN 388
LAMN 389
LAMN 390
LAMN 391

LAMNAR

PROFIL

PROF 001
 PROF 002
 PROF 003
 PROF 004
 PROF 005
 PROF 006
 PROF 007
 PROF 008
 PROF 009
 PROF 010
 PROF 011
 PROF 012
 PROF 013
 PROF 014
 PROF 015
 PROF 016
 PROF 017
 PROF 018
 PROF 019
 PROF 020
 PROF 021
 PROF 022
 PROF 023
 PROF 024
 PROF 025
 PROF 026
 PROF 027
 PROF 028
 PROF 029
 PROF 030
 PROF 031
 PROF 032
 PROF 033
 PROF 034
 PROF 035

```

SUBROUTINE PROFIL (IDSOOUT,LU)
  THIS ROUTINE WAS TAKEN FROM NASA TN D-5681 BY W.D. MCNALLY.
  COMMON /EXEC/CASE,TITLE,PAGE,ERROR
  COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,RENO,PFS,TFS,AFS,RHOFS,VIS
  COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEJ,TAPEK
  1 TAPEG,TAPEH,TAPEI,TAPEL,TAPEM,TAPEO,TAPEP,TAPEQ,TAPER,TAPE
  COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,CTYPE(2)
  INTEGER TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEJ,TAPEK
  1 TAPEG,TAPEH,TAPEI,TAPEL,TAPEM,TAPEO,TAPEP,TAPEQ,TAPER,TAPE
  COMMON/CI/ R,RTZ,TTZ,UPMACH,NST,NVP,NTURB,KPVM,KEM,KSMTH,
  1KSPN,KLE,KATCH,CINET,DLAM,TLAM,DTURB,TTURB,KPRE,KGRAD,KSCE,KLAM,
  2KMAIN,KPROF,X(100),Y(100),PRES(100),UE(100),ME(100),POPTZ(100),
  3VOVC(100),TVAL(100),
  4 PSZ,YSZ,UZ,ASZ,ATZ,RHSZ,RMTZ,MUSZ,MUTZ,NUSZ,NUTZ,CP,
  SPR,TC,ARCL,
  7 S(100),SQL(100),AE(100),TSE(100),
  8TAWL(100),TAWT(100),YBAR(100),RM(300),SM(100),SUTHL(100),
  9RMSW(100),RHSE(100),HEADW(100),HFADE(100),NUW(100),MUBAR(100),
  AAAA(100),BB(100),FF(100),DUDS(100),DMDR(100),DMDL(100),
  B THET(100),DELSR(100),DELTA(100),FOPM(100),
  CFORMI(100),FORMTR(100),RTH(100),RTHI(100),CF(100),
  DTAUM(100),NUSS(100),DTDY(100),HTRAN(100),CRN(100),
  E SHAPL(100),SHAPK(100),B,NS,
  F FTRAN,FORMS,
  G INST,I'FRAN,ISEP,
  H XTAB(103),YTAB1(103),YTAB2(103),NTAB,
  I EROR,TRANS,SEPRN
  REAL MF,NUSS
  C PRINT LOCATIONS OF INSTABILITY, TRANSITION, AND SEPARATION
  C IF (KMAIN,NE.1) GO TO 60
  
```

PROFIL

PROFIL

```

WRITE (TAPEOT,1000)
IF(I .EQ.0) GO TO 10
WRITE (TAPEOT,1010) INST
GO TO 20
10 WRITE (TAPEOT,1020)
20 IF (ITRAN.LE.1) GO TO 30
WRITE (TAPEUT,1030) ITRAN
GO TO 40
30 WRITE (TAPEOT,1040)
40 IF(ISEP.EQ.0) GO TO 50
WRITE (TAPEOT,1050) ISEP
GO TO 60
50 WRITE (TAPEUT,1060)

C PRINT LOCATIONS OF LAMINAR AND TURBULENT BOUNDARY LAYERS
C
60 IEND = ITRAN-1
IF (IEND.EQ.=1.OR.IEND.EQ.0) IEND=ISEP
IF (IEND.EQ.0) IEND=NST
IF (KMAIN.NE.1) GO TO 70
IF (ITRAN .EQ. 1) WRITE (TAPEOT,1070)
IF (ITRAN .NE. 1) WRITE (TAPEOT,1080) IEND
IF (ITRAN .EQ. 0) WRITE (TAPEOT,1090)
IF (ITRAN .EQ. 1) WRITE (TAPEOT,1100) ITRAN,IEND
70 IF (ITRAN.LE.1) GO TO 80
IEND = ISEP
IF(IEND.EQ.0) IEND=NST
IF (KMAIN.NE.1) GO TO 90
WRITE (TAPEOT,1100) ITRAN,IEND

C PRINT CALCULATED BOUNDARY LAYER PARAMETERS
C
80 IF (KMAIN.NE.1) GO TO 90
WRITE (TAPEOT,1110)
WRITE (TAPEOT,1120) (I,X(I),S(I),DELSK(I),THET(I),DELTA(I),

```

PROF 036
PROF 037
PROF 038
PROF 039
PROF 040
PROF 041
PROF 042
PROF 043
PROF 044
PROF 045
PROF 046
PROF 047
PROF 048
PROF 049
PROF 050
PROF 051
PROF 052
PROF 053
PROF 054
PROF 055
PROF 056
PROF 057
PROF 058
PROF 059
PROF 060
PROF 061
PROF 062
PROF 063
PROF 064
PROF 065
PROF 066
PROF 067
PROF 068
PROF 069
PROF 070

PROFIL

PROFIL

```

1  FORM(I),FORMI(I),I=1,IEND)
  WRITE (TAPEOT,1130)
  WRITE (TAPEOT,1140) (I,CF(I),TAUH(I),RTH(I),DTDY(I),NUSS(I),
  ) HTRAN(I),CRN(I),I=1,IEND)

C  COMPUTE BOUNDS ON VELOCITY PROFILES
C
90  IF (KPROF.NE.1) GO TO 170
  WRITE (TAPEOT,1150)
  IF (ITRAN.NE.0) GO TO 100
  IL1= 2
  IL2= IEND
  IT1= 0
  IT2= 0
  GO TO 110
100 IL1= 2
  IL2= ITRAN+1
  IT1= ITRAN
  IT2= IEND
  IF (IT1.EQ.1) IT1=2

C  CALCULATE AND PRINT LAMINAR BOUNDARY LAYER VELOCITY PROFILES
C
110 NVPI= NVPI+1
  IF (IL2.LT.IL1) GO TO 140
  DO 130 I=IL1,IL2
  WRITE (TAPEOT,1160) I
  * AA= 2.*SHAPL(I)/6.
  BB= .5*SHAPL(I)
  CCC= -2.*.5*SHAPL(I)
  DDD= 1.*SHAPL(I)/6.
  DEL= DELTA(I)/FLOAT(NVP)
  YP= -DEL
  DO 120 J=1,NVPI
  YP= YP+DEL

```

PROF 071
PROF 072
PROF 073
PROF 074
PROF 075
PROF 076
PROF 077
PROF 078
PROF 079
PROF 080
PROF 081
PROF 082
PROF 083
PROF 084
PROF 085
PROF 086
PROF 087
PROF 088
PROF 089
PROF 090
PROF 091
PROF 092
PROF 093
PROF 094
PROF 095
PROF 096
PROF 097
PROF 098
PROF 099
PROF 100
PROF 101
PROF 102
PROF 103
PROF 104
PROF 105

PROFIL

PROFIL

```
ETA= YP/DELTA(I)
YXMAX= YP/X(NST)
UUE=AAA*ETA+BBB*ETA**2+CCC*ETA**3+DDD*ETA**4
U= UUE*UE(I)
120 WRITE (TAPEOT,1180) ETA,YP,YXMAX,U,UUE
130 CONTINUE
C
C CALCULATE AND PRINT TURBULENT BOUNDARY LAYER VELOCITY PROFILES
C
140 IF(IT1.EQ.0) GO TO 170
DO 160 I=IT1,IT2
POWER= DELTA(I)/DELSR(I)-1.
WRITE (TAPEOT,1170) I,POWER
DELS= DELTA(I)/FLOAT(NVP)
YPS= -DEL
DO 150 J=1,NVPI
YPS= YP+DFL
EYAS= YP/DELTA(I)
YXMAX= YP/X(NST)
UUES= ETA**(1./POWER)
U= UUE*UE(I)
150 WRITE (TAPEOT,1180) ETA,YP,YXMAX,U,UUE
160 CONTINUE
C
C SAVE DELTA-STAR DATA FOR ITERATION IF REQUIRED
170 IF (IOSOUT .EQ. 0) GO TO 220
IF (IEND .EQ. NST) GO TO 190
DO 180 I=IEND,NST
180 DELSR(I) = DELSR(IEND)
C REVERSE ORDER OF DELTA-STAR ARRAY TO MATCH X-Y DATA ORDER IF THIS
C IS A LOWER SURFACE
190 IF (LU .EQ. 1) GO TO 210
IMAX = NST/2
N2 = NST + 1
DO 200 I=1,IMAX
```

PROF 106
PROF 107
PROF 108
PROF 109
PROF 110
PROF 111
PROF 112
PROF 113
PROF 114
PROF 115
PROF 116
PROF 117
PROF 118
PROF 119
PROF 120
PROF 121
PROF 122
PROF 123
PROF 124
PROF 125
PROF 126
PROF 127
PROF 128
PROF 129
PROF 130
PROF 131
PROF 132
PROF 133
PROF 134
PROF 135
PROF 136
PROF 137
PROF 138
PROF 139
PROF 140

PROFIL

PROFIL

```

11 = N2 - I
DELSAV = DELSR(II)
DELSR(II) = DELSR(I)
200 DELSR(I) = DELSAV
210 WRITE (IDSOUT) (DELSR(I),I=1,NSI)
220 RETURN
C
C  FORMAT STATEMENTS
C
1000 FORMAT(1M1//1X,36HPRINCIPAL BOUNDARY LAYER INFORMATION///)
1010 FORMAT (/10X,31HINSTABILITY OCCURS AT STATION ,I3)
1020 FORMAT (/10X,26HINSTABILITY DOES NOT OCCUR)
1030 FORMAT (/10X,30HTRANSITION OCCURS AT STATION ,I3)
1040 FORMAT (/10X,25HTRANSITION DOES NOT OCCUR)
1050 FORMAT (/10X,30HSEPARATION OCCURS AT STATION ,I3)
1060 FORMAT (/10X,25HSEPARATION DOES NOT OCCUR)
1070 FORMAT (/10X,37HLAMINAR BOUNDARY LAYER DOES NOT OCCUR)
1080 FORMAT (/10X,42HLAMINAR BOUNDARY LAYER = STATIONS 1 TO ,I3)
1090 FORMAT (/10X,39HTURBULENT BOUNDARY LAYER DOES NOT OCCUR///)
1100 FORMAT (/10X,35HTURBULENT BOUNDARY LAYER = STATIONS,2X,
113.6M TO ,I3//)
1110 FORMAT(/1X,7HSTATION,6X,1M,12X,1M,12X,5HDELSR,10X,4MTHET,11X,
15HDELTA,11X,4HFORM,10X,5HFORMI)
1120 FORMAT(2X,I3,3X,2F13.6,F14.6,1X,F14.6,1X,F14.6,1X,2F14.4)
1130 FORMAT(/1X,7HSTATION,6X,2HCF,13X,4MTAUM,11X,3HRTM,14X,4MDTDY,
113X,4MNUSS,10X,5HTRAM,12X,3HCRN)
1140 FORMAT(15,F14.5,2X,F14.5,1X,F12.1,5X,F14.2,2X,F14.2,1X,
1F14.4,2X,F13.3)
1150 FORMAT(1H1//1X,17HVELOCITY PROFILES//)
1160 FORMAT (/1X,7HSTATION,1X,15,2X,7HPROFILE/3X,7HY/DELTA,9X,
11HY,12X,6HY/XMAX,10X,1HU,12X,4HU/UE)
1170 FORMAT (/1X,7HSTATION,1X,15,2X,7HPROFILE,24X,2HNM,1X,F6,2/3X,7HY/D
1ELTA,9X,1HY,12X,6HY/XMAX,10X,1HU,12X,4HU/UE)
1180 FORMAT(1X,F8,4,2X,2G15.6,2X,F9.2,6X,F9.4)
END
PROF 141
PROF 142
PROF 143
PROF 144
PROF 145
PROF 146
PROF 147
PROF 148
PROF 149
PROF 150
PROF 151
PROF 152
PROF 153
PROF 154
PROF 155
PROF 156
PROF 157
PROF 158
PROF 159
PROF 160
PROF 161
PROF 162
PROF 163
PROF 164
PROF 165
PROF 166
PROF 167
PROF 168
PROF 169
PROF 170
PROF 171
PROF 172
PROF 173
PROF 174
PROF 175
```

PROFIL

TURBLN

TURB 001
 TURB 002
 TURB 003
 TURB 004
 TURB 005
 TURB 006
 TURB 007
 TURB 008
 TURB 009
 TURB 010
 TURB 011
 TURB 012
 TURB 013
 TURB 014
 TURB 015
 TURB 016
 TURB 017
 TURB 018
 TURB 019
 TURB 020
 TURB 021
 TURB 022
 TURB 023
 TURB 024
 TURB 025
 TURB 026
 TURB 027
 TURB 028
 TURB 029
 TURB 030
 TURB 031
 TURB 032
 TURB 033
 TURB 034
 TURB 035

```

SURROUTINE TURBLN
MODIFIED BY A.E. GENTRY

THIS ROUTINE WAS TAKEN FROM NASA TN D-5431 BY W.D. MCNALLY.

COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH, ALT, PSYAG, TBTAG, V, RENO, PFS, TFS, AFS, RHDFS, VIS
COMMON /TAPE/TAPEIN, TAPEOUT, TAPFA, TAPEB, TAPFC, TAPED, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 COMMON /GASD/GAS, GAGCP, PRAN, IGAS, AV1, AV2, AV3, GTYPE(2)
INTEGER TAPEIN, TAPEOUT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
1 COMMON/C1/ R, PTZ, ITZ, UPWACH, NST, NVP, NTURB, KPYM, KEM, KSMTM,
      KSPLN, KLE, KATCH, CTHET, DLAM, TLAM, DTURB, TURB, KPFE, KGRAD, KSDF, KLAM,
      ZKMAIN, KPROF, X(100), Y(100), PRFS(100), ME(100), PCPTZ(100),
      ZVOVCR(100), TVAL(100),
      PSZ, TSZ, UZ, ASZ, ATZ, RMSZ, RHTZ, MUSZ, MUTZ, NUSZ, NUTZ, CP,
      SPR, TC, ARCL,
      S(100), SOL(100), AE(100), TSE(100),
      TMT(100), TART(100), TBAR(100), RM(300), RM(100), SUTHL(100),
      9RHSW(100), RHSE(100), HEADW(100), HEADE(100), NUW(100), MUBAR(100),
      AAAA(100), BB(100), FF(100), DUDS(100), DMDE(100), DMOL(100),
      TMT(100), DELSR(100), DELTA(100), FORM(100),
      CFORHI(100), FORMTR(100), RTH(100), RTHI(100), CF(100),
      DTAUW(100), NUSS(100), DTOY(100), HTRAN(100), CRN(100),
      E SHAPL(100), SHAPK(100), B, NS,
      FTRAN, FORMS,
      INST, ITRAN, ISEP,
      H XTAB(103), YTAB1(103), YTAB2(103), NYAB,
      I ERROR, TRANS, SEPRN
REAL MUSZ, NUSZ, MUTZ, NUTZ, ME, MUW, MUBAR, NUSS
LOGICAL EKOR, TRANS, SFPRN

C SOLVE TURBULENT BOUNDARY LAYER DIFFERENTIAL EQUATIONS
C USING RUNGA-KUTTA
  
```

TURBLN

TURBLN

```

CALL RINKUT
*F (FROR) GO TO 40
*F (XSDE,NE.1) GO TO 10
WRITE (TAPEOT,1000)
WRITE (TAPEOT,1010) (XTAB(I),YTAB1(I),YTAB2(I),I=1,NTAB)

C CALCULATE TURBULENT BOUNDARY LAYER PARAMETERS AT EACH STATION
10 DO 30 I=ITRAN,NSI
IF (S(I).LE.XTAB,NTAB) GO TO 20
ISEP = I-1
SEPRN = .TRUE.
RETURN
20 TEM1 = 1.+5*(GAM-1.)+4E(I)**2
CALL LGRNGF(XTAB,YTAB1,NTAB,S(I),F)
THEYRE = NU7*F**7.686/ME(I)/ATZ
THEY(I) = THEYR*SQRT(YSE(I)/YTZ)*PIZ/PRES(I)
RYH(I) = UE(I)*THFT(I)/NUM(I)
CALL LGRNGF(XTAB,YTAB2,NTAB,S(I),FORMI(I))
FORMTR(I) = FORMI(I)*(1.+SU(I))
FORM(I) = FORMTR(I)*TFM1+PR**((1./3.)*(TFM1-1.))
DELSR(I) = THFT(I)*FORN(I)
POWER = 2.0/(FORMI(I)-1.0)
IF (FORMI(I).LT.1.02) POWER=100.
DELTA(I) = (1.+POWER)*DELSR(I)
CF(I) = 0.246*EXP(-1.561*FORMI(I))*F***(0.2113)
1 *TSE(I),TBAR(I)*(MUBAR(I)/MUY7)**0.268
TAUM(I) = CF(I)*HEADE(I)
IF (I.EG.1) GO TO 30
HTRAN(I) = CF(I)/2./PR**((2./3.)*RHSE(I)+UE(I)*CP*(TAWY(I)-TVAL(I)))
DTDY(I) = HTRAN(I)/TC
NUSS(I) = S(I)*DTDY(I)/(TAWY(I)-TVAL(I))
CRN(I) = CF(I)*RW(I)/NUSS(I)
30 CONTINUE
40 RETURN

```

TURBLN

TURBLN

```
1000 FORMAT(1H1//5X,62HTURBULENT DIFFERENTIAL EQUATIONS - SOLUTION FOR TURB 071
1 F AND FORMI///4(31H S FORMI )//) TURB 072
1010 FORMAT((4(F1.0,5,2X,F8.1,2X,F7.4,2X))) TURB 073
END TURE 074
```

TURBLN

RUNKUT

001 RUNK
002 RUNK
003 RUNK
004 RUNK
005 RUNK
006 RUNK
007 RUNK
008 RUNK
009 RUNK
010 RUNK
011 RUNK
012 RUNK
013 RUNK
014 RUNK
015 RUNK
016 RUNK
017 RUNK
018 RUNK
019 RUNK
020 RUNK
021 RUNK
022 RUNK
023 RUNK
024 RUNK
025 RUNK
026 RUNK
027 RUNK
028 RUNK
029 RUNK
030 RUNK
031 RUNK
032 RUNK
033 RUNK
034 RUNK
035 RUNK

SURROUTINE RUNKUT

C
C
C
C
C
C
C
C

RUNKUT SOLVES SIMULTANEOUS FIRST ORDER INITIAL VALUE
ORDINARY DIFFERENTIAL EQUATIONS

THIS ROUTINE WAS TAKEN FROM NASA TN D-5481 BY W.D. MCNALLY.

COMMON /EXEC/CASF, TITLE, PAGE, ERROR
COMMON /FLIGHT/MACH,ALT,PSTAG,YSTAG,V,RENO,PFS,TPS,AFS,RHDFS,VIS
COMMON /TAPE/TAPEIN,TAPECT,TAPEA,TAPEB,TAPEC,TAPFD,TAPEE,TAPEF,
1 TAPEG,TAPFH,TAPEI,TAPEJ,TAPEK
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
INTEGER TAPEIN,TAPECT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON/C1/ R,PYZ,YYZ,UPMACH,NST,NVP,NTURB,KPVM,KEM,KSMTM,
1KSPLN,KLF,KATCH,CTHET,DLAM,TLAM,DTURB,TTURB,KPRE,KGRAD,KSOF,KLAM,
2KMAIN,KPROF,X(100),Y(100),PRES(100),UE(100),ME(100),POPTZ(100),
3VOVCR(100),TVAL(100),
4 PSZ,TSZ,UZ,ASZ,ATZ,RMSZ,RMTZ,MUSZ,MUTZ,NUSZ,NUTZ,CP,
5PR,TC,ARCL,
7 S(100),SOL(100),AE(100),TSE(100),
8TAWL(100),TAMT(100),TBAR(100),RM(300),SW(100),SUTHL(100),
9RHSW(100),RHSE(100),HEADW(100),HEADE(100),NUW(100),MUBAR(100),
AAAA(100),BB(100),FF(100),DUDS(100),DMDR(100),DMOL(100),
B THEY(100),DELSR(100),DELTA(100),FORM(100),
CFORMI(100),FORMTR(100),RTH(100),RTHI(100),CF(100),
DTAUM(100),NUSS(100),DTDY(100),WTRAN(100),CRN(100),
E SHAPL(100),SHAPK(100),B,NS,
F FTRAN,FORMS,
G INST,I TRAN,ISEP,
H XTAB(103),YTAB1(103),YTAB2(103),NTAR,
I EROR,TRANS,SEPRN
LOGICAL EROR,TRANS,SEPRN

RUNKUT

RUNKUT

036 RUNK
 037 RUNK
 038 RUNK
 039 RUNK
 040 RUNK
 041 RUNK
 042 RUNK
 043 RUNK
 044 RUNK
 045 RUNK
 046 RUNK
 047 RUNK
 048 RUNK
 049 RUNK
 050 RUNK
 051 RUNK
 052 RUNK
 053 RUNK
 054 RUNK
 055 RUNK
 056 RUNK
 057 RUNK
 058 RUNK
 059 RUNK
 060 RUNK
 061 RUNK
 062 RUNK
 063 RUNK
 064 RUNK
 065 RUNK
 066 RUNK
 067 RUNK
 068 RUNK
 069 RUNK
 070 RUNK

```

C DIMENSION YY(2),RY(2),YINC(2),DUT(2),RUK(2,4)
C DOUBLE PRECISION XX,RX,YY,RY,RIK,DEL,DNT,
C 1TEM1,TEM2,TEM3,TEM4,TEM5,TEM6
C REAL ME,NUM,MUBAR
C FUNCY(X1,X1,X2,Y1,Y2) = (X1-X1)*(Y2-Y1)/(X2-X1) + Y1
C
C THE VALUE OF THE INCOMPRESSIBLE FORM FACTOR
C (FORMS, YY(2)) IS LIMITED TO THE FLAT PLATE
C VALUE. FORMFP. A VALUE OF 1.3 IS USED (SUGGESTED
C BY RESHOTKO + TUCKER, NACA TN-4154).
C FORMFP = 1.3
C
C SET DEL SPACING AND STORE INITIAL VALUES
C
C DEL = 0.002 * ARCL
C NSTEP = 0
C 10 YY(1) = PTRAN
C YY(2) = FORMS
C IF (YY(2) .LT. FORMFP) YY(2) = FORMFP
C XX = S(ITRAN)
C NV = 2
C NTAB = 1
C YTAB1(1) = YY(1)
C YTAB2(1) = YY(2)
C XTAB(1) = XX
C
C SOLVE FOR YY(1) AND YY(2) AT NEXT XX INCREMENT
C
C SAVE PREVIOUS YY(1) AND YY(2)
C 20 DO 30 J=1,NV
C 30 RY(J) = YY(J)
C RX = XX
C
C CALCULATE NEW YY(1) AND YY(2)
  
```

RUNKUT

RUNKUT

```

C
C   DN 90 L=1,4
C   PUT DIFFERENTIAL EQUATIONS IN THE FORM OF
C   FIRST DERIVATIVE = REMAINDER OF EQUATION
  IF (XX.LT. 3(2)) GO TO 35
  CALL LGRNGE(S,ME,NST,XX,ANS1)
  CALL LGRNGE(S,SW,NST,XX,ANS2)
  CALL LGRNGE(S,AAA,NST,XX,ANS3)
  CALL LGRNGE(S,BB,NST,XX,ANS4)
  CALL LGRNGE(S,DMDS,NST,XX,ANS5)
  CALL LGRNGE(S,TBAR,NST,XX,ANS6)
  GO TO 36
35 ANS1 = FUNCY (XX,S(1),S(2),ME(1),ME(2))
   ANS2 = FUNCY (XX,S(1),S(2),SW(1),SW(2))
   ANS3 = FUNCY (XX,S(1),S(2),AAA(1),AAA(2))
   ANS4 = FUNCY (XX,S(1),S(2),BB(1),BB(2))
   ANS5 = FUNCY (XX,S(1),S(2),DMDS(1),DMDS(2))
   ANS6 = FUNCY (XX,S(1),S(2),TBAR(1),TBAR(2))
36 CONTINUE
  TEM1 = 1.+(1.+(ANS2)*YY(2)
  TEM2 = 0.123D+0*DEXP(-1.561D+0*YY(2))*ANS3
  TEM3 = 0.123 * EXP(-1.561*YY(2))*ANS3
  DOT(1) = 1.268*(=YY(1)/AN31*ANS5*TEM1+TEM2)
  TEM3 = YY(2)*(YY(2)+1.)*2*(YY(2)-1.)
  TEM4 = 1.+(ANS2*(YY(2)*YY(2)+4.*YY(2)-1.)/((YY(2)+1.)*(YY(2)+3.))
  TEM5 = (YY(2)*YY(2)-1.)*YY(2)/YY(1)*TEM2
  IF (YY(1) .LE. 0.000) GO TO 110
  IF (YY(1) .LF. 0.0) GO TO 110
  TEM6 = (YY(2)*YY(2)-1.)/YY(1)**(.7886)*(.011*(YY(2)+1.)*(YY(2)+1.))
  1**2/YY(2)**2+TZ/ANS6)*ANS4
  DOT(2) = -ANS5*.5/ANS1*TEM3+TEM4+TEM5-TEM6
C   APPLY THE RUNGA-KUTTA SCHEME
  IF ((RX+DEL) .GT. S(NST)) DEL = S(NST) * RX
  DO 40 J=1,NV
  40 TUK(J,L) = DEL*DOT(J)

```

RUNK 071
RUNK 072
RUNK 073
RUNK 074
RUNK 075
RUNK 076
RUNK 077
RUNK 078
RUNK 079
RUNK 080
RUNK 081
RUNK 082
RUNK 083
RUNK 084
RUNK 085
RUNK 086
RUNK 087
RUNK 088
RUNK 089
RUNK 090
RUNK 091
RUNK 092C
RUNK 093
RUNK 094
RUNK 095
RUNK 096
RUNK 097I
RUNK 098C
RUNK 099
RUNK 100
RUNK 101
RUNK 102
RUNK 103
RUNK 104
RUNK 105

RUNKUT

RUNKUT

```

      GO TO (50,50,70,90), L
      50 DO 60 J=1,NV
      60 YY(J)=RY(J)+RUK(J,L)/2.
      IF (YY(2) .LT. FORMFP) YY(2) = FORMFP
      XX=RX+DEL/2.
      GO TO 65
      70 DO 80 J=1,NV
      80 YY(J)=RY(J)+RUK(J,L)
      IF (YY(2) .LY. FORMFP) YY(2) = FORMFP
      XX=RX+DEL
      85 IF (YY(2) .LE. 176.0D+0) GO TO 90
      85 IF (YY(2) .LE. 176.0) GO TO 90
      WRITE (TAPE0,86) YY(2)
      86 FORMAT (1H,18H**IN RUNKUT YY(2)=,F12.5,21H AND HAS BEEN SET TO ,
      1 30H176.0 TO PREVENT PROGRAM HALT. )
      C YY(2) = 176.0D+0
      C YY(2) = 176.0
      90 CONTINUE
      C INCREMENT THE DEPENDENT VARIABLES TO OBTAIN NEW YY(1) AND YY(2)
      DO 100 J=1,NV
      YINC(J) = (RUK(J,1)+2.*RUK(J,2)+2.*RUK(J,3)+RUK(J,4))'/6.
      100 YY(J)=RY(J)+YINC(J)
      IF (YY(2) .LT. FORMFP) YY(2) = FORMFP
      C CHECK FOR SEPARATION (HI .GT. 2.8)
      IF (YY(2) .LE. 2.6) GO TO 101
      YTAB1(NTAB+1) = YY(1)
      YTAB2(NTAB+1) = YY(2)
      XTAB(NTAB+1) = XX
      RETURN
      C STORE NEW COMPUTED VALUES IN A TABLE
      C
      101 NSTEP = N. EP + 1
      IF (NSTEP .NE. 5) GO TO 20
      NSTEP = 0

```

RUNK 106
RUNK 107
RUNK 108
RUNK 109
RUNK 110
RUNK 111
RUNK 112
RUNK 113
RUNK 114
RUNK 115
RUNK 116I
RUNK -17C
RUNK 118
RUNK 119
RUNK 120
RUNK 121I
RUNK 122C
RUNK 123
RUNK 124
RUNK 125
RUNK 126
RUNK 127
RUNK 128
RUNK 129
RUNK 130
RUNK 131
RUNK 132
RUNK 133
RUNK 134
RUNK 135
RUNK 136
RUNK 137
RUNK 138
RUNK 139
RUNK 140

RUNKUT

RUNKUT

```
NTAB = NTA5 + 1
YTAB1(NTAB) = YY(1)
YTAB2(NTAB) = YY(2)
YTAB(NTAB) = XX
IF (XX.LT.S(NST)) GO TO 20
RETURN
110 ERROR = .TRUE.
WRITE (TAPE0,120) L,YY(1),YY(2)
120 FORMAT (1H,48H**THE PARAMETER YY(J) IS .LE. 0.0 IN THE TEMP6 ,
C 1 13HEQUATION L=,I2,9H .,1),D18.10,9H YY(2)=,D18.10,71H ,
1 13HEQUATION L=,I2,9H YY(1)=,F10.5, 9H YY(2)=,F10.5,71H ,
2 32H **THIS CASE IS TERMINATED ** )
RETURN
END
```

RUNK 141
RUNK 142
RUNK 143
RUNK 144
RUNK 145
RUNK 146
RUNK 147
RUNK 148
RUNK 149
RUNK 150
RUNK 151
RUNK 152
RUNK 153
RUNK 154

RUNKUT

FUNCT

001 FUNC
 002 FUNC
 003 FUNC
 004 FUNC
 005 FUNC
 006 FUNC
 007 FUNC
 008 FUNC
 009 FUNC
 010 FUNC
 011 FUNC
 012 FUNC
 013 FUNC
 014 FUNC
 015 FUNC
 016 FUNC
 017 FUNC
 018 FUNC
 019 FUNC
 020 FUNC
 021 FUNC
 022 FUNC
 023 FUNC
 024 FUNC
 025 FUNC
 026 FUNC
 027 FUNC
 028 FUNC
 029 FUNC
 030 FUNC
 031 FUNC
 032 FUNC
 033 FUNC
 034 FUNC
 035 FUNC

FUNCT

```

SUBROUTINE FUNCT(XX,FX,DFX,INF)
C
C
C
C
      THIS ROUTINE WAS TAKEN FROM NASA TN D-54681 BY M.D. MCNALLY.

      COMMON /EXEC/CASE,ITITLE,PAGE,ERROR
      COMMON /FLIGHT/MACH,ALT,PSIAG,TSTAG,V,RFND,PFS,TF8,AFS,RHOF8,VIS
      COMMON /TAPE/TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPEE,TARFF,
1      TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
      COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
      INTEGER TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,
1      TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
      COMMON/CI/ R,PTZ,TTZ,UPMACH,NST,NVP,NTURB,KPVM,KEM,KSMTM,
1KSPLN,KLE,KATCH,CTHET,OLAM,TLAM,DTURB,TTURB,KPRE,KGRAD,KSDE,KLAM,
2KMAIN,KPROF,X(100),Y(100),PRES(100),UE(100),ME(100),POPTZ(100),
3VOYCR(100),TVAL(100),
4      PSZ,TSZ,UZ,ASZ,ATZ,RMSZ,RMTZ,MUSZ,MUTZ,NUSZ,NUTZ,CP,
5UR,TC,ARCL,
7      S(100),SOL(100),AE(100),TSE(100),
8TAWL(100),TAWT(100),TBAR(100),RM(300),SW(100),SUTHL(100),
9RHSW(100),RHSE(100),HEADW(100),HEADE(100),NUM(100),MUBAR(100),
AAA(100),BB(100),FF(100),DUDS(100),DMD(100),DMDL(100),
B      THET(100),DELSR(100),DELTA(100),FORM(100),
CFORMI(100),FORMTR(100),PTH(100),RTHI(100),CF(100),
DTAU(100),NUSS(100),DTDY(100),HTRAN(100),CRN(100),
E      SHAPL(100),SHAPK(100),B,NS,
F      RAN,FORMS,
G      INST,I TRAN,I REP,
H      XTAB(103),YTAB1(103),YTAB2(103),NTAB,
I      EROR,TRANS,SEPRN
      REAL MUSZ,NUSZ,MUTZ,NUTZ,ME,NUJW,MUBAR
      INF = 0
      R1 = 1.+5*(GAM-1.)*ME(1)**2
      R2 = 1.+(2.79+1.78*PR**5)*(1.+SW(1))*R1**1.
      R3 = NUTZ*SUTHL(1)*ARCL/ATZ/DMDL(1)*B1*(3.-GAM)/(2.*GAM-2.)
  
```

FUNCT

```
B4* =1.1138*B2  
B5= 2.38411*R2+(4.65*PR***(1./3.))-3.65*FR**5)*PR**5*(B1-1.)  
FX* (B3*XX)**5*(B4*XX+B5)  
IF (XX.EQ.0.) GO TO 10  
DFX* .5*(B3*XX)**(.,5)*B3*(B4*XX+B5)+B4*(B3*XX)**5  
RETURN  
10 INF = 1  
OFX = 1.E10  
RETURN  
END
```

FUNG 036
FUNG 037
FUNG 038
FUNG 039
FUNG 040
FUNG 041
FUNG 042
FUNG 043
FUNG 044
FUNG 045

FUNCT

INTI

INTI 001
 INTI 002
 INTI 003
 INTI 004
 INTI 005
 INTI 006
 INTI 007
 INTI 008
 INTI 009
 INTI 010
 INTI 011
 INTI 012
 INTI 013
 INTI 014
 INTI 015
 INTI 016
 INTI 017
 INTI 018
 INTI 019
 INTI 020
 INTI 021
 INTI 022
 INTI 023
 INTI 024
 INTI 025
 INTI 026
 INTI 027
 INTI 028
 INTI 029
 INTI 030
 INTI 031
 INTI 032
 INTI 033

```

REAL FUNCTION INTI(XX)
C
C THIS ROUTINE WAS TAKEN FROM NASA TN D-5661 BY W.D. MCNALLY.
C
COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,RENO,PFS,AFS,RHOPS,VIS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
COMMON/C1/R,PYZ,TTZ,UPMACH,NST,NVP,NTURB,KPVM,KEM,KSMTH,
1KSPLN,KLE,KATCH,CYHET,DLAM,TLAM,DTURB,YTURB,KPRF,KGRAD,KSDE,KLAM,
2KMAIN,KPROF,X(100),Y(100),PRES(100),UE(100),WE(100),POPTZ(100),
3VOVCR(100),TVAL(100),
4 PSZ,YSZ,UZ,ASZ,ATZ,RHSZ,RMTZ,MUSZ,MUTZ,NUSZ,MUTZ,CP,
5 SPR,TC,ARCL,
6 S(100),SIGL(100),AE(100),TSE(100),
7 CRAWL(100),TAWT(100),TBAR(100),RM(300),SM(100),SUTHL(100),
8 PRMSW(100),RHSE(100),HEADW(100),HEADE(100),NUM(100),MUBAR(100),
9 AAAA(100),BB(100),FF(100),DUDS(100),CHDS(100),DMDL(100),
10 THFT(100),DELSR(100),DELTA(100),FORM(100),
11 CFORMI(100),FORMTR(100),RTH(100),RTHI(100),CF(100),
12 DTAUM(100),MUSS(100),DTDY(100),MTRAN(100),CPN(100),
13 E SHAPL(100),SHAPK(100),B,NS,
14 FTRAN,FORMS,
15 INST,IITRAN,ISEP,
16 XSTAR(103),YTABI(105),YTAB2(103),NTAB,
17 EROR,TRANS,SEPRN
18 PEAL ME,NUM,MUBAR
19 CALL LGRNGE(SOL,ME,NST,XX,ANS)
20 CALL LGRNGE(SOL,AE,NST,XX,ANSAE)
21 CALL LGRNGE(SOL,PRES,NST,XX,ANSPE)
22 IF (ANS .LT. 0.0) ANS = ME(1) + 0.0001
23 INTI = ANS**(B+1.)*ANSAE*ANSPE/(ATZ*PI*7)
24 RETURN
25 END

```

INTI

C
C
C
C

INT2

INT2 001
 INT2 002
 INT2 003
 INT2 004
 INT2 005
 INT2 006
 INT2 007
 INT2 008
 INT2 009
 INT2 010
 INT2 011
 INT2 012
 INT2 013
 INT2 014
 INT2 015
 INT2 016
 INT2 017
 INT2 018
 INT2 019
 INT2 020
 INT2 021
 INT2 022
 INT2 023
 INT2 024
 INT2 025
 INT2 026
 INT2 027
 INT2 028
 INT2 029
 INT2 030
 INT2 031
 INT2 032
 INT2 033
 INT2 034
 INT2 035

INT2

```

REAL FUNCTION INT2(XX)
C
C THIS ROUTINE WAS TAKEN FROM NASA TN D-5481 BY W.D. McNALLY.
C
COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,FENO,PFS,TPS,AFS,RHOF5,VIS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AY2,AV3,GTYPE(2)
COMMON/C1/ R,PTZ,TTZ,UPMACH,NST,NVP,NTURB,KPVM,KEM,KSMTH,
1KSPLN,KLE,KATCH,CTHET,DLAM,TLAM,DTURB,TTURB,KPRE,KGRAD,KSE,KLAM,
2KMAIN,KPROF,X(100),Y(100),PRES(100),UE(100),WE(100),POPTZ(100),
3VOVCR(100),TVAL(100),
4 PSZ,TSZ,UZ,ASZ,ATZ,RHSZ,RHTZ,MUSZ,WUTZ,NUSZ,NUTZ,CP,
5 PR,TC,ARCL,
6 S(100),SOL(100),AE(100),TSE(100),
7 BTWL(100),TAMT(100),TBAR(100),RM(300),SM(100),SUTHL(100),
8PMSW(100),RHSE(100),HEADW(100),HEADE(100),NUW(100),MUBAR(100),
9AAA(100),BB(100),FF(100),DIJDS(100),DMDR(100),DMDL(100),
10 THET(100),DELSR(100),DELTA(100),FORM(100),
11 CFORNT(100),FORMTR(100),RTH(100),RTHI(100),CF(100),
12 DTAUM(100),NUSS(100),DTDY(100),HTRAN(100),CRN(100),
13 E SHAPL(100),SHAPK(100),B,NS,
14 F FTRAN,FORMS,
15 G INS,I,TRAN,ISEP,
16 H XTAB(103),YTABI(103),YTAB2(103),NTAB,
17 I EROR,TRANS,SEPRN
18 REAL ME,NUW,MURAR
19 IF (NS.LT.4) GO TO 10
20 CALL LGHNGE(SOL,SHAPK,NS,XX,INT2)
RETURN
21 DO 20 J=2,NS
22 IF (SOL(J),T,XX) GO TO 20
23 INT2=SHAPK(J=1)+(SHAPK(J)-SHAPK(J=1))*(XX-SOL(J=1))/(SOL(J)-SOL(J
24 1=1))
25 RETURN
26 CONTINUE

```

INT2

INT2 036
INT2 037

RETURN
END

INT2

LGRNGE

```

C
C SUBROUTINE LGRNGE(X,Y,N,ARG,ANS)
C LGRNGE PERFORMS 4 POINT LAGRANGIAN INTERPOLATION
C THIS ROUTINE WAS TAKEN FROM NASA TN D-5481 BY W.D. MCNALLY,
C
C DIMENSION X(N),Y(N),XX(4),YY(4)
C FUNCTY(XI,X1,X2,Y1,Y2) = (XI-X1)*(Y2-Y1)/(X2-X1) + Y1
C IF (N .LT. 4) GO TO 90
C IF (ARG=X(2)) 10,10,20
C 10 MM = 1
C GO TO 70
C 20 IF (ARG=X(N=1)) 40,40,30
C 30 MM = N-1
C GO TO 70
C 40 NI = N-1
C DO 60 I=2,NI
C IF (ARG=X(I)) 50,50,60
C 50 MM = I-2
C GO TO 70
C 60 CONTINUE
C 70 DO 80 I=1,4
C MM = MM+I-1
C XX(I) = X(MMM)
C YY(I) = Y(MMM)
C 80 C1 = ((ARG=XX(2))*(ARG=XX(3))*(ARG=XX(4)))/
C 1(YY(1)=XX(2))/(XX(1)=XX(3))/(XX(1)=XX(4))
C C2 = ((ARG=XX(1))*(ARG=XX(3))*(ARG=XX(4)))/
C 1(XX(2)=XX(1))/(XX(2)=XX(3))/(XX(2)=XX(4))
C C3 = ((ARG=XX(1))*(ARG=XX(2))*(ARG=XX(4)))/
C 1(XX(3)=XX(1))/(XX(3)=XX(2))/(XX(3)=XX(4))
C C4 = ((ARG=XX(1))*(ARG=XX(2))*(ARG=XX(3)))/
C 1(XX(4)=XX(1))/(XX(4)=XX(2))/(XX(4)=XX(3))
C ANS = C1*YY(1)+C2*YY(2)+C3*YY(3)+C4*YY(4)
C RETURN

```

LGRN 001
 LGRN 002
 LGRN 003
 LGRN 004
 LGRN 005
 LGRN 006
 LGRN 007
 LGRN 008
 LGRN 009
 LGRN 010
 LGRN 011
 LGRN 012
 LGRN 013
 LGRN 014
 LGRN 015
 LGRN 016
 LGRN 017
 LGRN 018
 LGRN 019
 LGRN 020
 LGRN 021
 LGRN 022
 LGRN 023
 LGRN 024
 LGRN 025
 LGRN 026
 LGRN 027
 LGRN 028
 LGRN 029
 LGRN 030
 LGRN 031
 LGRN 032
 LGRN 033
 LGRN 034
 LGRN 035

LGRNGE

LGRN5E

LGRN 036
LGRN 037
LGRN 038
LGRN 039
LGRN 040
LGRN 041
LGRN 042
LGRN 043
LGRN 044
LGRN 045

```
C
C LESS THAN 4 POINTS * USE LINEAR INTERPOLATION
90 IF (ARG = X(2)) 110,100,120
100 ANS = Y(2)
    GO TO 130
110 ANS = FUNCY(ARG,X(1),X(2),Y(1),Y(2))
    GO TO 150
120 ANS = FUNCY(ARG,X(2),X(3),Y(2),Y(3))
150 RETURN
END
```

LGRN5E

ROOT

```

SUBROUTINE ROOT(A,B,Y,FUNCT,TOLERY,X,DFX)
C ROOT FINDS A ROOT FOR (FUNCT-Y) IN THE INTERVAL (A,B)
C THIS ROUTINE WAS TAKEN FROM NASA TN D-55881 BY W.D. MCNALLY.
COMMON /TAPE/TAPEIN,TAPEOUT,TAPEA,TAPEE,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
1 INTEGER TAPEIN,TAPEOUT,TAPEA,TAPEB,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
X1= A
X2= B
CALL FUNCT(X1,FX1,DFX,INF)
10 DO 30 I=1,20
X= (X1+X2)/2,
CALL FUNCT(X,FX,DFX,INF)
IF ((FX1-Y)*(FX-Y).GT.0.) GO TO 20
X2= X
GO TO 30
20 X1= X
FX1= FX
30 CONTINUE
IF (ABS(Y-FX).LT.TOLERY) RETURN
WRITE (TAPEOUT,1000) A,B,Y
STOP
1000 FORMAT(//2X,4SHRROOT HAS FAILED TO CONVERGE IN THE GIVEN INTERVAL
1/4X,3HA B,G14.6,10X,3HB B,G14.6,10X,3HY B,G14.6)
END

```

ROOT 001
ROOT 002
ROOT 003
ROOT 004
ROOT 005
ROOT 006
ROOT 007
ROOT 008
ROOT 009
ROOT 010
ROOT 011
ROOT 012
ROOT 013
ROOT 014
ROOT 015
ROOT 016
ROOT 017
ROOT 018
ROOT 019
ROOT 020
ROOT 021
ROOT 022
ROOT 023
ROOT 024
ROOT 025
ROOT 026
ROOT 027
ROOT 028

ROOT

CURVFT

```
C
C SUBROUTINE CURVFT(COEF,ANS,X,Y,NX,NY)
C EVALUATE THE POLYNOMIAL FUNCTION, ANS=F(X,Y), USING COEFFICIENTS, COE
C
C THIS ROUTINE WAS TAKEN FROM NASA TN D-5001 BY W.D. MCNALLY.
C
      DIMENSION COEF(20)
      NX1 = NX+1
      NY1 = NY+1
      ANS = COEF(1)
      IF (X.EQ..0.AND.Y.EQ..0) RETURN
      IF (Y.EQ..0) GO TO 10
      IF (X.EQ..0) GO TO 30
      GO TO 50
10 DO 20 I=2,NX1
20 ANS = ANS+COEF(I)*X**(I-1)
   RETURN
30 DO 40 J=2,NY1
40 ANS = ANS+COEF(K)*Y**(J-1)
   RETURN
50 ANS = 0
   DO 60 J=1,NY1
   DO 60 J=1,NX1
   K = (I-1)*NX1+J
60 ANS = ANS+COEF(K)*Y**(I-1)*X**(J-1)
   RETURN
   END
```

CURV 001
CURV 002
CURV 003
CURV 004
CURV 005
CURV 006
CURV 007
CURV 008
CURV 009
CURV 010
CURV 011
CURV 012
CURV 013
CURV 014
CURV 015
CURV 016
CURV 017
CURV 018
CURV 019
CURV 020
CURV 021
CURV 022
CURV 023
CURV 024
CURV 025
CURV 026
CURV 027
CURV 028

CURVFT

GRADNT

```

C
C
C
C
C
C
SUBROUTINE GRADNT(X,FX,N,DFDX)
GRADNT CALCULATES THE GRADIENT OF FX WITH RESPECT TO X
USING FINITE DIFFERENCE TECHNIQUES
THIS ROUTINE WAS PREPARED BY D.N. SMYTH
DIMENSION X(N),FX(N),DFDX(N)
DIMENSION SL(2),DIST(2)
N1=N-1
SL(1)=(FX(2)-FX(1))/(X(2)-X(1))
DFDX(1)=SL(1)
DIST(1)=SQRT((FX(2)-FX(1))**2+(X(2)-X(1))**2)
DO 20 I=2,N1
SL(2)=(FX(I+1)-FX(I))/(X(I+1)-X(I))
DIST(2)=SQRT((FX(I+1)-FX(I))**2+(X(I+1)-X(I))**2)
DFDX(I)=(SL(2)+DIST(1)+SL(1)*DIST(2))/(DIST(1)+DIST(2))
SL(1)=SL(2)
DIST(1)=DIST(2)
20 CONTINUE
SL(1)=(FX(N1)-FX(N1+1))/(X(N1)-X(N1+1))
DIST(1)=SQRT((FX(N1)-FX(N1+1))**2+(X(N1)-X(N1+1))**2)
DFDX(N)=(SL(2)+SL(2)+DIST(2))/(DIST(2)+DIST(1))
RETURN
END

```

```

GRAD 001
GRAD 002
GRAD 003
GRAD 004
GRAD 005
GRAD 006
GRAD 007
GRAD 008
GRAD 009
GRAD 010
GRAD 011
GRAD 012
GRAD 013
GRAD 014
GRAD 015
GRAD 016
GRAD 017
GRAD 018
GRAD 019
GRAD 020
GRAD 021
GRAD 022
GRAD 023
GRAD 024
GRAD 025

```

GRADNT

SMTMNA

SMTM 001
 SMTM 002
 SMTM 003
 SMTM 004
 SMTM 005
 SMTM 006
 SMTM 007
 SMTM 008
 SMTM 009
 SMTM 010
 SMTM 011
 SMTM 012
 SMTM 013
 SMTM 014
 SMTM 015
 SMTM 016
 SMTM 017
 SMTM 018
 SMTM 019
 SMTM 020
 SMTM 021
 SMTM 022
 SMTM 023

```

SUBROUTINE SMTMNA (Y,N,K)

C
C THIS ROUTINE WAS PREPARED BY D.N. SMYTH
C
C SMOOTH IS A SIMPLE DATA SMOOTHING ROUTINE.
C IF K=1, THE END POINTS ARE ALSO SMOOTHED.
C
    DIMENSION Y(N)
    N1 = N - 1
    Y1 = Y(1)
    Y2 = Y(2)
    Y3 = Y(3)
    IF (K .EQ. 1) Y(1) = (Y1 + 2.*Y2 + Y3) / 2.0
    DO 10 I=2,N1
    Y(I) = (Y1 + 2.*Y2 + Y3) / 4.0
    Y1 = Y2
    Y2 = Y3
    Y3 = Y(I+1)
    10 CONTINUE
    IF (K .EQ. 1) Y(N) = (Y3 + 2.*Y2 + Y1) / 2.0
    RETURN
    END
  
```

SMTMNA

SIMPS1

SIMP 001
 SIMP 002
 SIMP 003
 SIMP 004
 SIMP 005
 SIMP 006
 SIMP 007
 SIMP 008
 SIMP 009
 SIMP 010
 SIMP 011
 SIMP 012
 SIMP 013
 SIMP 014
 SIMP 015
 SIMP 016
 SIMP 017
 SIMP 018
 SIMP 019
 SIMP 020
 SIMP 021
 SIMP 022
 SIMP 023
 SIMP 024
 SIMP 025
 SIMP 026
 SIMP 027
 SIMP 028
 SIMP 029
 SIMP 030
 SIMP 031
 SIMP 032
 SIMP 033
 SIMP 034
 SIMP 035

```

FUNCTION SIMPS1(X1,X2,FUNC,KSIG)
C THIS ROUTINE CALCULATES THE INTEGRAL OF FUNC USING SIMPSONS RULE
C BETWEEN LIMITS X1 AND X2, TO BE LESS THAN A
C SPECIFIED TOLERANCE (EPS).
C THIS ROUTINE WRITTEN BY D. N. SMYTH.
C
C DATA EPS,NMAX/1.0E-5, 400/
C M IS THE NUMBER OF INTERVALS, H IS THE INTERVAL SPACING.
C
N = 2
DX = X2 - X1
H = DX/M
K = N/2
X = X1
C = FUNC(X)
ANS = 0.0
DO 30 I = 1,K
  A = C
  X = X + H
  B = FUNC(X)
  X = X + H
  C = FUNC(X)
  ANS = ANS + H*(A + 4.*B + C)
30 CONTINUE
C
C IF (K = 2) 31,32,33
C
31 RI = ANS
N = 4
  
```

SIMP81

SIMPS1

8IMP 036
 8IMP 037
 8IMP 038
 8IMP 039
 8IMP 040
 8IMP 041
 8IMP 042
 8IMP 043
 8IMP 044
 8IMP 045
 8IMP 046
 8IMP 047
 8IMP 048
 8IMP 049
 8IMP 050
 8IMP 051
 8IMP 052
 8IMP 053
 8IMP 054
 8IMP 055
 8IMP 056

```

C C
      GO TO 10
      R1 = ANS
      32 F = (ANS - R1)/(15.*ANS)
      IF (ABS(F) .LE. EPS) GO TO 60
      H = H*(EPS/ABS(F)).**0.25
      IF (H .LT. 0.0025*DX) H = 0.0025*DX
      N = 0.5*DX/H
      N = (N + 1)**2
      IF (N .GT. NMAX) N = NMAX
      GO TO 10
      33 E = (ANS - R1)/(15.*ANS)
      60 KSIG = 0
      SIMPS1 = ANS*(1.0 + E)/3.0
      RETURN
      END
C C
C C
  
```

SIMPS1

SPLINE

```

SUBROUTINE SPLINE(X,Y,N,H,D2YDX2)
C
C SPLINE FITS A SPLINE CURVE TO X AND Y
C AND CALCULATES FIRST AND SECOND DERIVATIVES AT THE SPLINE POINTS
C END POINT SECOND DERIVATIVES EQUAL THOSE AT ADJACENT POINTS
C
      DIMENSION X(N),Y(N),H(N),D2YDX2(N)
      DIMENSION G(100)
      G(1)= -1.
      H(1)= 0.
      N1= N-1
      IF (N1.LT.2) GO TO 20
      DO 10 I=2,N1
      A= (X(I)-X(I-1))/6.
      B= (X(I+1)-X(I))/6.
      C= 2.*(A+B)-A*B
      D= (Y(I+1)-Y(I))/(X(I+1)-X(I))-Y(I)-Y(I-1))/(X(I)-X(I-1))
      G(I)= B/C
      10 H(I)= (C-A*(I-1))/C
      20 D2YDX2(N)= H(N1)/(1.+G(N1))
      DO 30 I=2,N
      K= N+1-I
      30 D2YDX2(K)= H(K)-G(K)+D2YDX2(K+1)
      H(1) = (X(1)-X(2))/6.*(2.+D2YDX2(1)+D2YDX2(2))+(Y(2)-Y(1))/(X(2)
1-X(1))
      DO 40 I=2,N
      40 H(I) = (X(I)-X(I-1))/6.*(2.+D2YDX2(I)+D2YDX2(I+1))+(Y(I)-Y(I-1))
1/(X(I)-X(I-1))
      RETURN
      END

```

SPLI 001
SPLI 002
SPLI 003
SPLI 004
SPLI 005
SPLI 006
SPLI 007
SPLI 008
SPLI 009
SPLI 010
SPLI 011
SPLI 012
SPLI 013
SPLI 014
SPLI 015
SPLI 016
SPLI 017
SPLI 018
SPLI 019
SPLI 020
SPLI 021
SPLI 022
SPLI 023
SPLI 024
SPLI 025
SPLI 026
SPLI 027
SPLI 028
SPLI 029
SPLI 030

SPLINE

SPLINE

```

C
C SUBROUTINE SPLINE(X,Y,N,H,D2YDX2)
C
C SPLINE FITS A SPLINE CURVE TO X AND Y
C AND CALCULATES FIRST AND SECOND DERIVATIVES AT THE SPLINE POINTS
C END POINT SECOND DERIVATIVES EQUAL THOSE AT ADJACENT POINTS
C
DIMENSION X(N),Y(N),H(N),D2YDX2
DIMENSION G(100)
G(1) = -1.
H(1) = 0.
N1 = N-1
IF (N1.LT.2) GO TO 20
DO 10 I=2,N1
  A = (X(I)-X(I-1))/6.
  B = (X(I+1)-X(I))/6.
  C = 2.*(A*B) = A*B
  D = (Y(I+1)-Y(I))/(X(I+1)-X(I)) = (Y(I)-Y(I-1))/(X(I)-X(I-1))
  G(I) = B/C
10 H(I) = (D-A+H(I-1))/C
20 D2YDX2(N) = H(N1)/(1.+G(N1))
DO 30 I=2,N
  K = N+1-I
30 D2YDX2(K) = H(K) = G(K) = D2YDX2(K+1)
  H(1) = (X(1)-X(2))/6. + (2.*D2YDX2(1) + D2YDX2(2)) / (X(2)
    1 - X(1))
DO 40 I=2,N
40 H(I) = (X(I)-X(I-1))/6. + (2.*D2YDX2(I) + D2YDX2(I+1)) / (Y(I)-Y(I-1))
  1/(X(I)-X(I-1))
RETURN
END

```

```

SPLI 001
SPLI 002
SPLI 003
SPLI 004
SPLI 005
SPLI 006
SPLI 007
SPLI 008
SPLI 009
SPLI 010
SPLI 011
SPLI 012
SPLI 013
SPLI 014
SPLI 015
SPLI 016
SPLI 017
SPLI 018
SPLI 019
SPLI 020
SPLI 021
SPLI 022
SPLI 023
SPLI 024
SPLI 025
SPLI 026
SPLI 027
SPLI 028
SPLI 029
SPLI 030

```

SPLINE

SPEC

```

OVERLAY (MARK4,2,5)
PROGRAM SPEC
SUBROUTINE SPEC

COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPEOT, TAPEA, TAPE3, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
INTEGER ERROR, PAGE
INTEGER TAPEIN, TAPEOT, TAPEA, TAPE3, TAPEC, TAPEE, TAPEF,
1 TAPEG, TAPEH, TAPEI, TAPEJ, TAPEK
2 DIMENSION TITLE(15), IPG(20)
3 READ , TAPEIN, IO) IPG
10 FORMAT (20I1)

DO 80 I=1,20
  IPRG = IPG(I)
  IF (IPRG .EQ. 0) GO TO 90
  GO TO (30,40), IPRG

5 FORCE DATA SUMMATION OPTION
30 CALL SUM
  GO TO 80

C TRIA OPTION (TO BE ADDED LATER)
40 CONTINUE
  GO TO 80

C 80 CONTINUE

C 90 RETURN
  90 CONTINUE
  END

```

SPEC 001C
SPEC 002C
SPEC 003I
SPEC 004
SPEC 005
SPEC 006
SPEC 007
SPEC 008
SPEC 009
SPEC 010
SPEC 011
SPEC 012
SPEC 013
SPEC 014
SPEC 015
SPEC 016
SPEC 017
SPEC 018
SPEC 019
SPEC 020
SPEC 021
SPEC 022
SPEC 023
SPEC 024
SPEC 025
SPEC 026
SPEC 027
SPEC 028
SPEC 029
SPEC 030
SPEC 031
SPEC 032
SPEC 033

SPEC

SUM

SUM	001
SUM	002
SUM	003
SUM	004
SUM	005
SUM	006
SUM	007
SUM	008
SUM	009
SUM	010
SUM	011
SUM	012
SUM	013
SUM	014
SUM	015
SUM	016
SUM	017
SUM	018
SUM	019
SUM	020
SUM	021
SUM	022
SUM	023
SUM	024
SUM	025
SUM	026
SUM	027
SUM	028
SUM	029
SUM	030
SUM	031
SUM	032
SUM	033
SUM	034
SUM	035

SUM

```

SUBROUTINE SUM
C THIS OPTION IS USED TO SUM UP SELECTED FORCE DATA SETS. IT ALSO
C MAY BE USED TO PUNCH SELECTED SETS OF FORCE DATA.
C
COMMON /XFC/CASE,TITLE,PAGE,ERROR
COMMON /TAPE/TAPEIN,TAPEOT,TAPEA,TAPEB,TAPEC,TAPEE,TAPEF,TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
COMMON /FLIGHT/MACH,ALT,PSTAG,TSTAG,V,RENO,PFS,TFS,AFS,RHOPS,VIS
COMMON /GASD/GAM,GASCP,PRAN,IGAS,AV1,AV2,AV3,GTYPE(2)
COMMON /REF/SREF,MAC,SPAN,XCG,YCG,ZCG
C
DIMENSION E1(8),E2(11),E4(41),F(11),FF(11),E5(41),E6(41),
1 IPANL(10),TPAN9(10,20),LSUM(41),LCOM(41),E7(11),ICOMP(20),
2 FR(14)
DIMENSION TITLE(15)
C
INTEGER ERROR,PAGE,CASE
INTEGER TAPFIN,TAPEOT,TAPEI,TAPEJ,TAPEK,TAPEC,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
REAL MACH,MAC
DATA TITLE9/4MHFORC/
DATA TITLE9/10HFORCE DATA/
DATA BLANK/4H /
C
DO 5 I = 1,15
5 TITLE(I) = BLANK
CASE = 0
C
C READ MASTER TABLE OF CONTENTS FOR UNIT 9
IG9 = 1
CALL READMS (9,E1,8,IG9)
READ (9,IG9) E1
C NUMBER OF COMPONENTS ON UNIT
NTOT = E1(1)

```

SUM

```

C   NUMBER OF SUMMATIONS ON UNIT
      NSUM = E1(2)
C   NEXT EMPTY RECORD
      NEXT = E1(3)
C   CHECK IF UNIT HAS VALID DATA
      IF (E1(4) = E9. TITLE9) GO TO 20
      WRITE (TAPE0,10)
10   FORMAT (1M0,47M**UNIT 9 CALLED BY ROUTINE SUM HAS NEVER BEEN ,
1     27INITIALIZED, PROGRAM STOP, )
      STOP
C   20 CONTINUE
C
C   IG9 = 4
      CALL READMS (9,E8,1-,IG9)
      READ (9#IG9) E8
      MACH = E8(1)
      V = E8(2)
      REND = E8(3)
      ALT = E8(4)
      PSTAG = E8(5)
      TSTAG = E8(6)
      GTYPE(1) = E8(7)
      GTYPE(2) = E8(8)
      SRFF = E8(9)
      SPAN = E8(10)
      MAC = E8(11)
      XCG = E8(12)
      YCG = E8(13)
      ZCG = E8(14)
C   POINTERS TO FORCE DATA
C   IG9 = 2
      CALL READMS (9,E5,41,IG9)
      READ (9#IG9) E5

```

SUM 036
SUM 037
SUM 038
SUM 039
SUM 040
SUM 041
SUM 042
SUM 043
SUM 044
SUM 045
SUM 046
SUM 047
SUM 048
SUM 049
SUM 050C
SUM 051I
SUM 052
SUM 053
SUM 054
SUM 055
SUM 056
SUM 057
SUM 058
SUM 059
SUM 060
SUM 061
SUM 062
SUM 063
SUM 064
SUM 065
SUM 066
SUM 067
SUM 068
SUM 069C
SUM 070I

SUM

SUM

00 30 1e2,41
 30 LCOM(I-1) = E5(I)
 IFTYPE = E5(I)
 I69 = 3
 CALL READMS (9,E6,41,I69)
 READ (9MIG9) E6
 07 60 1e2,41
 40 LSUM(I-1) = E6(I)
 I8TYPE = E6(I)
 00 41 1e1,11
 41 E7(I) = 0.0
 READ SUMMATION CONTROL CARD
 ISUM = 0 (DO NOT SUM)
 I1 = 1 (SUM COMPONENTS)
 IPUNCH = 0 (DO NOT PUNCH)
 I1 (PUNCH SUMMATIONS ONLY)
 I2 (PUNCH COMPONENTS PLUS SUMMATIONS)
 IPRINT = 0 (DO NOT PRINT COMPONENTS, ONLY PRINT FINAL SUMMATION)
 I1 (PRINT COMPONENTS PLUS SUMMATION)
 50 READ (TAPEIN,55) LAST,ISUM,ISAVE,IPRINT,IPUNCH,IRUN,ICOMP
 55 FORWARD (5:1,I4,20I2)
 IF (ISUM .GT. 0) NSUM = NSUM + 1
 IF (ISUM .GT. 0) LSUM(NSUM) = NEXT
 NEXTS = NEXT
 NSUMS = NSUM
 NEXT = NEXT + 2
 ORIGIN ALPHA=BETA ARRAY FOR FIRST COMPONENT TO BE READ
 = ICOMP(I)
 I69 = LCOM(L) + 1
 CALL READMS (9,E4,41,I69)

SUM 071
 SUM 072
 SUM 073
 SUM 074
 SUM 075
 SUM 076C
 SUM 077I
 SUM 078
 SUM 079
 SUM 080
 SUM 081
 SUM 082
 SUM 083
 SUM 084
 SUM 085
 SUM 086
 SUM 087
 SUM 088
 SUM 089
 SUM 090
 SUM 091
 SUM 092
 SUM 093
 SUM 094
 SUM 095
 SUM 096
 SUM 097
 SUM 098
 SUM 099
 SUM 100
 SUM 101
 SUM 102
 SUM 103
 SUM 104
 SUM 105C

SUM

SUM

```

C READ (9*IG9, EA
  NAB = F4(1)
SUM 106I
SUM 107
SUM 108
SUM 109
SUM 110
SUM 111
SUM 112
SUM 113
SUM 114
SUM 115
SUM 116
SUM 117C
SUM 118I
SUM 119
SUM 120
SUM 121
SUM 122
SUM 123
SUM 124
SUM 125
SUM 126
SUM 127
SUM 12A
SUM 129
SUM 13A
SUM 131
SUM 132
SUM 133
SUM 136
SUM 135
SUM 136
SUM 137
SUM 138C
SUM 139I
SUM 140
C DO LOOP FOR COMPONENTS SELECTED
  K = 0
  50 K = K + 1
  COMPONENT NUMBER
  L = ICOMP(K)
  IF (L .EQ. 0) GO TO 155
  NCT = K
  IGR = LCOM(L)
  CALL READMS (9,E2,I1,IG9)
  READ (9*IG9) R2
  LL = E2(1)
  IF (LL .EQ. L) GO TO 100
  WRITE (TAPE9,60)
  60 FORMAT (1H0,45H**COMPONENT NUMBER REQUESTED DOES NOT MATCH ,
    1 47#NUMBER ON UNIT 9 IN ROUTINE SUM, PROGRAM STOP, )
  STOP
C RECALL PANEL NUMBERS
  100 DO 110 I=1,10
    IPAN9(I,K) = E2(I+1)
  110 CONTINUE
C PRINT OR PUNCH COMPONENTS AS REQUIRED
  IPR = 0
  IF (IPRINT.EQ.0 .AND. IPUNCH.NE.2) GO TO 150
  IF (IPUNCH .EQ. 2) WRITE (TAPE9,121) L,
  121 FORMAT (26#COMPONENT DATA SET NUMBER ,I3)
  IF (IPRINT .EQ. 1) GO TO 195
  122 DO 124 I=1,NAB
    IGR = LCOM(L) + I*2
    CALL READMS (9,F,I1,IG9)
    READ (9*IG9) F
  C IF (IPRINT .EQ. 1) WRITE (TAPE9,290) IF

```

SUM

SUM

```

141 IF (IPUNCH ,EQ. 2) WRITE (TAPE6,123) F(1),F(6),F(4),F(9),F(5),
SUM
142 F(11),F(10),F(7),IRUN
SUM
143 FORMAT (7F9.4,7,2,1X,14)
SUM
144 CONTINUE
SUM
145 IF (IPUNCH ,EQ. 2) WRITE (TAPE1,125)
SUM
146 FORMAT (1X ,5X,33MCOMPONENT DATA HAVE BEEN PUNCHED, )
SUM
147 IF (K .LT. 20) GO TO 58
SUM
148 CONTINUE
SUM
149 IPP = 1
SUM
150 IF (ISUM ,EQ. 0) GO TO 440
SUM
151
SUM
152 DO LOOP FOR ALL ALPHA=BETA VALUES
SUM
153 DO 190 J=1,NAF
SUM
154 ZERO OUT FORCE ARRAY
SUM
155 DO 160 I=1,11
SUM
156 F(7) = 0.0
SUM
157
SUM
158 DO 180 K=1,NDCY
SUM
159 L = ICOMP(K)
SUM
160 IG9 = LCOM(L) + J*2
SUM
161 CALL READMS (9,FF,11,IG9,
SUM
162C READ (9#IG9) FF
SUM
163I DO 170 I=2,11
SUM
164 F(I) = F(I) + FF(I)
SUM
165
SUM
166 170 CONTINUE
SUM
167 180 CONTINUE
SUM
168
SUM
169 IF (F(2) ,EQ. 0.0) F(2) = 0.000000
SUM
170 F(R) = F(3) / F(2)
SUM
171 F(1) = FF(1)
SUM
172 F(7) = FF(7)
SUM
173
SUM
174 CALL ARITHMS (9,F,11,NEXT)
SUM
175C WRITE (9#NEXT) F
SUM
176I

```

SUM

SUM

```

NEXT = NEXT + 1
CALL WRITMS (9,E7,11,NEXT)
WRITE (9#NEXT) E7
NEXT = NEXT + 1
190 CONTINUE
C
C
C
C***OUTPUT DATA *****
C WRITE HEADER INFORMATION FOR FORCE OUTPUT PAGE
195 CALL HEADER
IF (IPR .EQ. 0) WRITE (TAPE0,198) L
198 FORMAT (1H0,25HCOMPONENT DATA SET NUMBER ,I3)
IF (IPR .NE. 1) GO TO 204
WRITE (TAPE0,200) NSUM
200 FORMAT (1H0,26HSUMMATION DATA SET NUMBER ,I3)
DO 202 I=1,NCT
WRITE (TAPE0,203) (IPAN9(YI,I),I=1,10)
202 CONTINUE
203 FORMAT (1H ,5X,10I4)
204 WRITE (TAPE0,230) MACH,V,RENU
230 FORMAT (1H0,7H MACH#F8,3,6H VEL#F9,1,16H FT/SEC RE/FT #E13.5 )
IF (PSTAG .LT. 0.00001) WRITE (6,240) ALT
240 FORMAT (1H ,7H ALT #F8,0 )
IF (PSTAG .GT. 0.00001) WRITE (6,250) PSTAG,TSTAG
250 FORMAT (1H ,16X7HP STAG#F7,1,16H ATMOS T STAG#F7,1,6H DEG F )
WRITE (TAPE0,260) GTYPE
260 FORMAT (1H ,2X,2A4)
WRITE (TAPE0,270) SREF,SPAN,MAC,XCG,YCG,ZCG
270 FORMAT (1H0,9H S REF #F9,2,8H SPAN #F8,2,8H MAC #F8,2,71H ,
1 9H X CG #F9,2,8H Y CG #F8,2,8H Z CG #F8,2 )
WRITE (TAPE0,280)
280 FORMAT (1H0,10HFORCE DATA,71H ,7H ALPHA,4X,3HC D,7X,3HC L,7X,
1 3HC A,7X,3HC Y,7X,3HC N,71H ,7H BETA ,4X,3HL,C,7X,3HC M,7X,
2 4HC LL,6X,4HC LN)

```

SUM 176
SUM 177C
SUM 178I
SUM 179
SUM 180
SUM 181
SUM 182
SUM 183
SUM 184
SUM 185
SUM 186
SUM 187
SUM 188
SUM 189
SUM 190
SUM 191
SUM 192
SUM 193
SUM 194
SUM 195
SUM 196
SUM 197
SUM 198
SUM 199
SUM 200
SUM 201
SUM 202
SUM 203
SUM 204
SUM 205
SUM 206
SUM 207
SUM 208
SUM 209
SUM 210

SUM

SUM

```

IF (IPR .EQ. 0) GO TO 122
IF (IPUNCH .NE. 0) WRITE (TAPEG,285) NSUM
285 FORMAT (27H$SUMMATION COMPONENT NUMBER ,I3)
C
C RETRIEVE OUTPUT DATA FROM UNIT 9 AND PRINT
DO 300 I=1,NAB
  IG9 = LSUM(NSUM) + I*2
  CALL READMS (9,F,11,IG9)
  REAL (9#IG9) F
C
C WRITE OUTPUT FORCE DATA
  WRITE (TAPEOT,290) F
290 FORMAT (1H0,F7.2,5F10.5,/1H ,F7.2,4F10.5)
  IF (IPUNCH .NE. 0) WRITE (TAPEG,123) F(1),F(6),F(4),F(9),F(5),
    1 F(11),F(10),F(7),IRUN
300 CONTINUE
  IF (IPUNCH .NE. 0) WRITE (TAPEOT,302)
302 FORMAT (1H ,5X,25HYESE DATA WHERE PUNCHED. )
C**OUTPUT DATA COMPLETED FOR THIS COMPONENT
C
C IF (ISAVE .NE. 0) NEXT = NEXTS
  IF (ISAVE .NE. 0) NSUM = NSUMS
C
C SAVE COMPONENT INFORMATION IN TABLE OF CONTENTS
  IF (ISAVE .NE. 0) GO TO 440
  DO 191 I=2,11
    E2(I) = ICOMP(I-1)
    E2(1) = NSUM
    IG9 = LSUM(NSUM)
    CALL WRITMS (9,E2,11,IG9)
    WRITE (9#IG9) F2
    IG9 = IG9 + 1
    CALL WRITMS (9,E4,41,IG9)
    WRITE (9#IG9) E4
  C
  C UPDATE MAIN TABLE OF CONTENTS

```

SUM 211
SUM 212
SUM 213
SUM 214
SUM 215
SUM 216
SUM 217
SUM 218C
SUM 219I
SUM 220
SUM 221
SUM 222
SUM 223
SUM 224
SUM 225
SUM 226
SUM 227
SUM 228
SUM 229
SUM 230
SUM 231
SUM 232
SUM 233
SUM 234
SUM 235
SUM 236
SUM 237
SUM 238
SUM 239C
SUM 240I
SUM 241
SUM 242C
SUM 243I
SUM 244
SUM 245

SUM

SUM

SUM	246
SUM	247
SUM	248
SUM	249
SUM	250C
SUM	251I
SUM	252
SUM	253
SUM	254
SUM	255
SUM	256C
SUM	257I
SUM	258
SUM	259
SUM	260
SUM	261
SUM	262
SUM	263
SUM	264
SUM	265
SUM	266
SUM	267
SUM	268
SUM	269

SUM

```

F1(1) = NTOY
E1(2) = NSUM
E1(3) = NEXT
IG9 = 1
CALL WRITMS (9,E1,8,109)
WRITE (9#IG9) E1
DO 310 I=2,41
  310 E6(I) = LSUM(I#1)
      E6(1) = 0,0
      IG9 = 3
CALL WRITMS (9,E6,41,IG9)
WRITE (9#IG9) E6
C
C
C
      WRITE (TAPEOT,430) NSUM
      430 FORMAT (1H0,10X,37HDATA SAVED ON UNIT 9, SUMMATION SET ,
      1 8HNUMBER =,14)
C
C
C
      CHECK IF LAST CASE HAS BEEN COMPLETED
      440 IF (LAST.EQ. 0) GO TO 50
C
      RETURN
      END

```


STREAM

STRE 036
 STRE 037
 STRE 038
 STRE 039
 STRE 040
 STRE 041
 STRE 042
 STRE 043
 STRE 044
 STRE 045
 STRE 046
 STRE 047
 STRE 048
 STRE 049
 STRE 050
 STRE 051
 STRE 052
 STRE 053
 STRE 054
 STRE 055
 STRE 056
 STRE 057C
 STRE 058I
 STRE 059
 STRE 060C
 STRE 061I
 STRE 062
 STRE 063
 STRE 064
 STRE 065
 STRE 066
 STRE 067
 STRE 068
 STRE 069
 STRE 070

```

C INPUT REGION ID INFO AND NORMALIZATION FLAGS
10 READ (TAPEIN,20) LASTR,NDSET,IABSET,IR,INORM,ISURF,IPF,
   1 (ISR(I),I=1,10), ISF
20 FORMAT (I1,3I2,3I1, 10I2,5I1)
C READ BOUNDARY POINTS FOR NORMALIZATION
DO 38 I=1,4
38 READ (TAPEIN,50) X6(I),YR(I),ZR(I)
50 FORMAT (3F10.0)
25 READ (TAPEIN,25) IRSAVE,NSTR,TITLER
25 FORMAT (2I2,16X,10A4)

C WRITE (TAPEOT,32) LASTR,NDSET,IABSET,I=9, INORM,ISURF,IRSAVE,
   1 NSTR,IPF, (ISR(I),I=1,10), ISF, TITLER
32 FORMAT (1H0,5X,6HLASTR=,I2,4X,6HNDSET=,I2,4X,7HIABSET=,I2,4X,
   1 4HIR =,I2, 4X,6HINORM=,I2,4X,6HISURF=,I2,4X,
   2 7HIRSAVE=,I2,4X,5HNSTR=,I2,/1M ,4HIPF=,I2,4X4HISR=, 10I3,
   3 4X4HISF=,5I2,/1H0, 5X7HTITLER=, 10A4)

C
C
C READ IN MASTER DIRECTORY
   ITAG10 = 1
   CALL READMS (10,TITLFM,10,ITAG10)
   READ(10,ITAG10) TITLER
   ITAG10 = 2
   CALL READMS (10,IMTAB,9,ITAG10)
   READ(10,ITAG10) IMTAB

C CHECK DATA SET NUMBER
   IF (IMTAB(1) .GE. NDSET) GO TO 40
   WRITE (TAPEOT,39) NDSET,IMTAB(1)
39 FORMAT (1H0,9HNDSET =,I2,3IM IS GREATER THAN THE NUMBER OF ,
   1 4GHDATA SETS ACTUALLY ON UNIT 10. PROGRAM HALT. )
   STOP
C
C
  
```

STREAM

STREAM

```
C READ TN FLOW DATA SET DIRECTORY
  40 ITAG10 = INTAR(NDSFT + 4)
  CALL READMS (10,E4,12,ITAG10)
  READ (10:ITAG10) E4
  DO 41 I=1,10
    41 TTTLES(I) = F4(I)
    MACH = E4(11)
    NABS = E4(12)
    ITAG10 = ITAG10 + 1
    CALL READMS (10,LOAD,20,ITAG10)
    READ(10:ITAG10) LOAD
C
C READ TN FLOW REGION DIRECTORY FOR REQUESTED ALPHA=BETA SET
  ITAG10 = LOAD(IABSET)
  CALL READMS (10,E2,13,ITAG10)
  READ (10:ITAG10) E2
  DO 42 I=1,10
    42 TTTLEA(I) = F2(I)
    ALPHAS = E2(11)
    BETAS = E2(12)
    NREG = E2(13)
    ITAG10 = ITAG10 + 1
    CALL READMS (10,LORG,20,ITAG10)
    READ(10:ITAG10) LORG
C
C CHECK IF THE STREAMLINE REGION WILL WIFF OUT SOME
  PREVIOUSLY SAVED REGION DIRECTORY
  IF (IRSAVE .EQ. 0) GO TO 52
  IF (IRSAVE .GT. NREG) GO TO 52
  WRITE (TAPEOT,51) IRSAVE,NREG
  51 FORMAT (1H0,39H**STREAMLINE SAVE REGION NUMBER INPUT (,I2,
  2 1 52H) IS LESS THAN THE NUMBER OF REGIONS ALREADY ON THE ,/1H ,
  2 22HALPHA=BETA DIRECTORY (,I2,25H). TO AVOID DESTROYING A ,
  3 57HPREVIOUSLY SAVED REGION DIRECTORY THE JOB WILL BE HALTED.)
```

STRE 071
STRE 072
STRE 073C
STRE 074I
STRE 075
STRE 076
STRE 077
STRE 078
STRE 075
STRE 080C
STRE 081I
STRE 082
STRE 083
STRE 084
STRE 085C
STRE 086I
STRE 087
STRE 088
STRE 089
STRE 090
STRE 091
STRE 092
STRE 093C
STRE 094I
STRE 095
STRE 096
STRE 097
STRE 098
STRE 099
STRE 100
STRE 101
STRE 102
STRE 103
STRE 104
STRE 105

STREAM

STREAM

STRE 141
 STRE 142
 STRE 143
 STRE 144
 STRE 145
 STRE 146
 STRE 147
 STRE 148
 STRE 149
 STRE 150
 STRE 151
 STRE 152
 STRE 153
 STRE 154
 STRE 155
 STRE 156
 STRE 157
 STRE 158
 STRE 159
 STRE 160
 STRE 161
 STRE 162
 STRE 163
 STRE 164
 STRE 165
 STRE 166
 STRE 167
 STRE 168
 STRE 169
 STRE 170
 STRE 171
 STRE 172
 STRE 173
 STRE 174
 STRE 175

COSPS = COS(PHIO*RC)
 SINS = SIN(PHIO*RC)
 SINP = SIN(PHIO*RC)
 COSP = COS(PHIO*RC)
 AP11 = COST*COSSP
 AP12 = COST*SINS
 AP13 = SINT
 AP21 = -COSP*SINS + SINP*SINT*COSSP
 AP22 = COSP*COSSP + SINP*SINT*SINS
 AP23 = SINP*COST
 AP31 = SINP*SINS + COSP*SINT*COSSP
 AP32 = -SINP*COSSP + COSP*SINT*SINS
 AP33 = COSP*COST

C SET INDICES FOR NORMALIZING DATA
 IF(INORM .LT. 0) GO TO 500
 GO TO (510, 540, 550), INORM

C NORMALIZE W.R.T. A,R
 500 IX = 4
 IY = 5
 IZ = 6
 GO TO 590

C NORMALIZE W.R.T. X,Y
 510 IX = 1
 IY = 2
 IZ = 3
 GO TO 590

C NORMALIZE W.R.T. X,Z
 530 IX = 1
 IY = 3
 IZ = 2
 GO TO 590

STREAM

STREAM

STRE 176
 STRE 177
 STRE 178
 STRE 179
 STRE 180
 STRE 181
 STRE 182
 STRE 183
 STRE 184
 STRE 185
 STRE 186
 STRE 187
 STRE 188
 STRE 189
 STRE 190
 STRE 191
 STRE 192
 STRE 193
 STRE 194
 STRE 195
 STRE 196
 STRE 197
 STRE 198
 STRE 199
 STRE 200
 STRE 201
 STRE 202
 STRE 203
 STRE 204
 STRE 205
 STRE 206
 STRE 207
 STRE 208
 STRE 209
 STRE 210

```

C      NORMALIZE W,R,T:  Y,Z
580 IX = 2
    IY = 3
    IZ = 1
    GO TO 590

C      NORMALIZE W,R,T.  A,PHI
550 IX = 4
    IY = 6
    IZ = 5
    GO TO 590

C      CALCULATE NORMALIZING LENGTHS
590 CONTINUE
    DO 595 I = 1,4
      XX =XB(I) * XO
      YY =YB(I) * YO
      ZZ =ZB(I) * ZO
      XP(I,1) = XX*AP11 + YY*AP12 + ZZ*AP13
      XP(I,2) = XX*AP21 + YY*AP22 + ZZ*AP23
      XP(I,3) = XX*AP31 + YY*AP32 + ZZ*AP33
      XP(I,4) = XP(I,1)
      XP(I,5) = SQRT(XP(I,2)**2 + XP(I,3)**2)
      IF (XP(I,2) .EQ. 0.0) GO TO 594
      XP(I,6) = ATAN2(XP(I,2), -XP(I,3))
      GO TO 595
594 XP(I,6) = 0.0
      IF (XP(I,3) .GT. 0.0) XP(I,6) = 3.141592654
595 CONTINUE

C      U1 = XP(1,IX)
      UL = XP(2,IX) * U1
      CR = UL

```

STREAM

STREAM

STRE 211
 STRE 212
 STRE 213
 STRE 214
 STRE 215
 STRE 216
 STRE 217
 STRE 218
 STRE 219
 STRE 220
 STRE 221
 STRE 222
 STRE 223
 STRE 224
 STRE 225
 STRE 226
 STRE 227
 STRE 228
 STRE 229
 STRE 230
 STRE 231
 STRE 232
 STRE 233
 STRE 234
 STRE 235
 STRE 236
 STRE 237
 STRE 238
 STRE 239
 STRE 240
 STRE 241
 STRE 242
 STRE 243
 STRE 244
 STRE 245

STREAM

```

IF (ISURF .EQ. 1) GO TO 596
V1 = XP(3,IV)
VL = XP(4,IV) = V1
GO TO 597

C
596 UL = XP(3,IX)
CT = XP(4,IX) = XP(3,IX)
V1 = XP(1,IV)
VL = XP(3,IV) = V1
597 CONTINUE

C
C
C
C
C
C ESTABLISH SUB-REGION COUNTERS
IF (ISR(1) .GT. 0) GO TO 310
NS = NSREG
DO 300 I = 1,NS
300 ISR(I) = I
GO TO 330
310 NS = 0
DO 320 I = 1,10
IF (ISR(I) .LE. 0) GO TO 330
320 NS = NS + 1

C
330 IF (NS .LE. NSREG) GO TO 350
WRITE(TAPEOT,340) NS,NSREG
340 FORMAT(1H0,23HSURFACE INTERPOLATION, THE ,I2,13H SUB-REGIONS ,
1 36HREQUESTED IS GREATER THAN AVAILABLE./1H0, 12HCOUNTER SET ,
2 11HTO NSREG = , I2, 20H AND CASE CONTINUES.)

C
NS = NSREG
350 CONTINUE
C
C

```

STREAM

STRE 246
STRE 247
STRE 248
STRE 249
STRE 250
STRE 251
STRE 252
STRE 253
STRE 254
STRF 255
STF 256
ST 257
STRE 258
STRE 259
STRE 260
STRE 261
STRE 262
STRE 263
STRE 264
STRE 265
STRE 266
STRE 267
STRE 268
STRE 269
STRE 270
STRE 271
STRE 272
STRE 273
STRE 274
STRE 275
STRE 276
STRE 277
STRE 278
STRE 279
STRE 280

```
I = 1
ISFR = 0
IF (IPF .EQ. 0) GO TO 400
ISFR = ISF(I)
IF (ISFR .GT. 5) ISFR = 0

C CALCULATE COEFFICIENTS FOR INTERPOLATION
400 CONTINUE
CALL SFNTRP

C IF (ERROR .NE. 0) GO TO 1000

C INITIALIZE NEW REGION DIRECTORY FOR STREAMLINE DATA
IF (IRSAVE .EQ. 0) GO TO 49
LCNEXT = INTAB(2)

C SET POINTERS
DO 163 I=1,5
  LOFF(I) = 0
  LOCO(I) = 0
  LOSF(I) = 0
  IPC(I) = 0
  IPD(I) = 0
  IDTYP(I) = 0
163 E3(I+10) = IDTYP(I)
  IDTYP(1) = 3
  IDTYP(2) = 2
  ITFLAG = 0
DO 48 I=1,10
  E3(I) = TITLER(I)
  E3(11) = 3
  E3(12) = 2
  E3(17) = ITFLAG
48 I610 = LCNEXT + NSTR + 4
  LOGC(IRSAVE) = LCNEXT

C
```

STREAM

STREAM

```
C SET STRFAMLINE COUNTER
49 ISRS = 0
C
C READ STREAMLINE DATA CARD
55 READ (TAPEIN,60) IPRINT,ISAVE,ISTART,ISTAG,ISMODE,
1 IPANL,L,DELTA3,XSI,YSI,ZSI
60 FORMAT (2I2,3I1,I2,I4,7X,4F10.0)
IF (IRSAVE .EQ. 0) GO TO 61
LOFF(1) = IG10 + 1
DO 59 I=1,12
59 DATA(I) = 0,0
C
61 ISRS = ISRS + 1
IF (ISRS .GT. 1) WRITE (TAPEOT,149)
149 FORMAT (1MI)
WRITE (TAPEOT,126) ISRS,IPRINT,ISAVE,ISTART,ISTAG,ISMODE,
1 IPANL,L,DELTA3,XSI,YSI,ZSI
126 FORMAT (1H0,1RSTREAMLINE NUMBER ,I2, /1H ,3X,7HIPRINT,13,4X,
1 6MISAVE,12,4X,7HISTART,12,4X,6HISTAG,12,4X,7HISMODE,12, /1H ,
2 3X,7HIPANL =,13,4X,3HL =,14,4X,7HDELTA3=,F10.0,4X,4HYSI=,F10.0,
3 4X,4HYSI=,F10.0,4X,4HZSI=,F10.0)
LINE = 0
IF (ISTART .NE. 0) GO TO 90
C
C OBTAIN POINTERS AND READ STARTING POINT ELEMENT DATA FROM UNIT 4
IG4 = 2
CALL READMS (4,E,25,IG4)
READ (4,IG4) E
NREM = E(4)
IG4 = IPANL + 5
CALL READMS (4,E,25,IG4)
READ (4,IG4) E
ISTART = E(3) = NREM
IG4 = ISTART + NREM + L
CALL READMS (4,E,25,IG4)
```

STRE 281
STRE 282
STRE 283
STRE 284
STRE 285
STRE 286
STRE 287
STRE 288
STRE 289
STRE 290
STRE 291
STRE 292
STRE 293
STRE 294
STRE 295
STRE 296
STRE 297
STRE 298
STRE 299
STRE 300
STRE 301
STRE 302
STRE 303
STRE 304
STRE 305
STRE 306
STRE 307C
STRE 308I
STRE 309
STRE 310
STRE 311C
STRE 312I
STRE 313
STRE 314
STRE 315C

STREAM

STREAM

STRE 3161
 STRE 317
 STRE 318
 STRE 319
 STRE 320
 STRE 321
 STRE 322
 STRE 323
 STRE 324
 STRE 325
 STRE 326
 STRE 327
 STRE 328
 STRE 329
 STRE 330
 STRE 331
 STRE 332
 STRE 333
 STRE 334
 STRE 335
 STRE 336
 STRE 337
 STRE 338
 STRE 339
 STRE 340
 STRE 341
 STRE 342
 STRE 343
 STRE 344
 STRE 345
 STRE 346
 STRE 347
 STRE 348
 STRE 349
 STRE 350

```

C   READ (4#164) E
    LL = E(1)
    IF (LL .EQ. L) GO TO 70
    WRITE (TAPEOT,60) L,LL
    60  FORMAT (1M0,2I#INPUT ELEMENT NUMBER ,14,
    1  45M IS NOT EQUAL TO THE ELEMENT NUMBER ON UNIT 4,14)
        STOP
    70  XS = E(7)
        YS = E(8)
        ZS = E(9)
        GO TO 110

C   USE INPUT STARTING POINT LOCATION
C   90 IF (ISTART .NE. 1) GO TO 100
        XS = XSI
        YS = YSI
        ZS = ZSI

C   IF ISMODE = 1 DETERMINE ZS FROM INTERPOLATION
C   IF ISMODE = 2 DETERMINE YS FROM INTERPOLATION
C   (ADD LATER)
C   GO TO 110
C   OBTAIN STARTING POINT FROM PREVIOUSLY CALCULATED STREAMLINE
C   100 CONTINUE

C   (ADD LATER)
C   (ADD LATER)
C   110 CONTINUE
        IF (ISTAG .EQ. 0) GO TO 120

C   CALCULATE STAGNATION POINT
  
```

STREAM

STREAM

```

C STEP RUNGE-KUTTA CYCLE COUNTER
130 IS = IS + 1
C
C INTERPOLATE FOR VARIABLES
  (ISMODE ,NE. 0) GO TO 131
  CALL VALUE (1,X,Y,Z,A(1),A(2),A(3),IC,IT,INT,
  1   EMT,EMX,EMY,EMZ,POP,TOT)
  IC = 1
  GO TO 132
131 CALL VALUE (1,A(1),A(2),A(3),X,Y,Z,IC,IT,INT,
  1   EMT,EMX,EMY,EMZ,POP,TOT)
C
C CHECK IF COORDINATES WHERE WITHIN BOUNDS
132 CONTINUE
  IF (INT(1) .EQ. 1) GO TO 140
C
C COORDINATES WHERE NOT WITHIN BOUNDS
  WRITE (TAPEOT,133)
133 FORMAT ('H ,47HNEXT COORDINATE POINT IS OUTSIDE NORMALIZATION ,
  1 11BOUNDARIES. ')
C
C
  GO TO 230
C
C SET DERIVATIVE ARRAY WITH THE INTERPOLATED DIRECTION COSINES
C OF THE SURFACE VELOCITY VECTOR
140 C(1) = EMX
  C(2) = EMY
  C(3) = EMZ
C
C CHECK IF WE ARE AT THE END OF A DELTAS CYCLE
  IF (IS ,NE. 1) GO TO 170
  IF (ISMODE ,NE. 0) GO TO 141
  XH = X

```

STREAM

STREAM

```
YW = Y
ZW = Z
GO TO 142
141 XW = A(1)
    YW = A(2)
    ZW = A(3)

C CHECK PRINT FLAG AND PRINT IF NECESSARY
C 142 IF (IPRINT .EQ. 0) GO TO 160
    IP = IP + 1
    IF (IP .NE. IPRINT) GO TO 160
    IP = 0
    WRITE (TAPEOT,150) S,XW,YW,ZW,EMT,PCP,TOT
150 FORMAT (1H,7F12.4)
    LINE = LINE + 1
    IF (LINE .GE. 50) WRITE (TAPEOT,149)
    IF (LINE .GE. 50) WRITE (TAPEOT,125)
    IF (LINE .GE. 50) LINE = 0

C
C
C CHECK IF STREAMLINE DATA ARE TO BE SAVED ON UNIT 10
160 IF (IRSAVE .EQ. 0) GO TO 170
    ISCT = ISCT + 1
    IF (ISCT .NE. ISAVE) GO TO 170
    ISCT = 0
    DATA(1) = XW
    DATA(2) = YW
    DATA(3) = ZW
    DATA(4) = S
    DATA(7) = EMT
    DATA(11) = POP
    DATA(12) = TOT
    IG10 = IG10 + 1
    CALL WRITMS (10,DATA,12,IG10)
    WRTTF (10=IG10) DATA

C
```

STRE 421
STRE 422
STRE 423
STRE 424
STRE 425
STRE 426
STRE 427
STRE 428
STRE 429
STRE 430
STRE 431
STRE 432
STRE 433
STRE 434
STRE 435
STRE 436
STRE 437
STRE 438
STRE 439
STRE 440
STRE 441
STRE 442
STRE 443
STRE 444
STRE 445
STRE 446
STRE 447
STRE 448
STRE 449
STRE 450
STRE 451
STRE 452
STRE 453
STRE 454C
STRE 455I

STREAM

STREAM

STRE 456
STRE 457
STRE 458
STRE 459
STRE 460
STRE 461
STRE 462
STRE 463
STRE 464
STRE 465
STRE 466
STRE 467
STRE 468
STRE 469
STRE 470
STRE 471
STRE 472
STRE 473
STRE 474
STRE 475
STRE 476
STRE 477
STRE 478
STRE 479
STRE 480
STRE 481
STRE 482
STRE 483
STRE 484
STRE 485
STRE 486
STRE 487
STRE 488
STRE 489
STRE 490

```

C      ICT = ICT + 1
C
C      170 GO TO (180,190,200,210), IS
C
C      FIRST RUNGE-KUTTA CYCLE
180 D = S
    DO 185 I=1,M
    F(I) = A(I)
    G(4*I-3) = C(I)*DELTA
185 A(I) = F(I) + G(4*I-3) / 2.0
    GO TO 190
C
C      SECOND RUNGE-KUTTA CYCLE
190 DO 195 I=1,M
    G(4*I-2) = C(I)*DELTA
195 A(I) = F(I) + G(4*I-2) / 2.0
197 S = D + DELTA/2.0
    GO TO 220
C
C      THIRD RUNGE-KUTTA CYCLE
200 DO 205 I=1,M
    G(4*I-1) = C(I) + DELTA
205 A(I) = F(I) + G(4*I-1)
    S = D + DELTA
    GO TO 220
C
C      FOURTH RUNGE-KUTTA CYCLE
210 DO 215 I=1,M
    G(4*I) = C(I) + DELTA
    A(I) = G(4*I-3) + 2.0*(G(4*I-2)+G(4*I-1))
215 A(I) = (A(I) + G(4*I)) / 6.0 + F(I)
    IS = 0
C
```

STREAM

STREAM

```

220 CONTINUE
C   END OF RUNGE-KUTTA CODE
    GO TO 130
C
C
C   SET SUBREGION POINTERS IN REGION DIRECTORY IF DATA WHERE SAVED
230 IF (IRSAVE .EQ. 0) GO TO 235
    IF (ICT .GT. 0) GO TO 233
    WRITE (TAPEOT,232)
232  FORMAT (1H0.46H**NN STREAMLINE POINTS WHERE CALCULATED. DUMP )
    I = 100000
    IFC(T) = 0
    STOP
233  IFC(1) = ICT
    DO 231 I=1,5
        DATB(I) = LOFF(I) + 0.001
        DATP(I+5) = IFC(I) + 0.001
        DATB(I+10) = LOCD(I) + 0.001
        DATB(I+15) = IFC(I) + 0.001
        DATB(I+20) = LOSF(I) + 0.001
        ITAG10 = LORG(IRSAVE) + ISRS + 4
        CALL WRITMS (10,DATB,25,ITAG10)
        WRITE (10,ITAG10) DATB
C   CHECK IF LAST STREAMLINE HAS BEEN REACHED
235  IF (ISRS .LT. NSTR) GO TO 240
C
    IF (IRSAVE .EQ. 0) GO TO 240
    E3(16) = NSTR
    ITAG10 = LORG(IRSAVE)
    CALL WRITMS (10,E3,17,ITAG10)
    WRITE (10,ITAG10) E3
C   STORE PLANE ORIENTATION DATA (SAME AS FOR SURFACE DATA)
    ITAG10 = ITAG10 + 1
    CALL WRITMS (10,DAT,6,ITAG10)
    WRITE (10,ITAG10) DAT
C

```

STREAM

STREAM

```
C RE-SET ALPHA-BETA DIRECTORY POINTERS IF DATA WERE SAVED
IF (IRSAVE .EQ. 0) GO TO 240
ITAG10 = LOAR(IARSET)
E2(13) = NREG
CALL WRITMS (10,E2,13,ITAG10)
WRITE (10,ITAG10) E2
ITAG10 = ITAG10 + 1
CALL WRITMS (10,LORG,20,ITAG10)
WRITE (10,ITAG10) LORG
C
C
C CHECK IF LAST REGION HAS BEEN USED
240 IF (LASTR .EQ. 0) GO TO 10
C RE-SET MASTER DIRECTORY POINTERS IF DATA WERE SAVED
IF (IRSAVE .EQ. 0) GO TO 900
IMTAB(2) = IG10 + 1
ITAG10 = 2
CALL WRITMS (10,IMTAB,9,ITAG10)
READ (10,ITAG10) IMTAB
GO TO 900
C
1000 WRITE (TAPEOT,1010)
1010 FORMAT (1H,43M**ERROR DETECTED AFTER RETURN FROM SFNTRP,1
1 17H PROGRAM STOP,*** )
STOP
C
C 900 RETURN
C 900 CONTINUE
END
```

STRE 526
STRE 527
STRE 528
STRE 529
STRE 530
STRE 531C
STRE 532I
STRE 533
STRE 534C
STRE 535I
STRE 536
STRE 537
STRE 538
STRE 539
STRE 540
STRE 541
STRE 542
STRE 543
STRE 544C
STRE 545I
STRE 546
STRE 547
STRE 548
STRE 549
STRE 550
STRE 551
STRE 552
STRE 553I
STRE 554C
STRE 555

STREAM

SFNTRP

```
C
C
SUBROUTINE SFNTRP
COMMON /EXEC/CASE, TITLE, PAGE, ERROR
COMMON /TAPE/TAPEIN, TAPECT, TAPEA, TAPEB, TAPEC, TAPEE, TAPEF,
      TAPEG, TAPEH, TAPEI, TAPEJ, TAPFK
1 COMMON/SURENT/INORM, ISURF, IX, IY, IZ, U1, U2, V1, V2, CR, CT, CHX, NP, N3,
2 FLOWC(7), XI(500), YI(500), B(503,7), XHF(503), AA(14400),
A XR(4), YB(4), ZB(4),
3 N0SET, I0BSET, IR, L0RG(20), ISR(20), NS, JSFR, IYI3, IFTYP, IFLOW,
4 BX(2,20), BY(2,20), NB
C
C DIMENSION TITLE(15), F4(12), E4B(12), E1(25), LOSF(5),
1 FLOW(503, 7),
      XT(6), VT(6), BP(2)
C
C INTEGER ERROR, PAGE, CASE
C INTEGER TAPEIN, TAPECT, TAPEA, TAPEB, TAPFC, TAPEE, TAPEF,
1 TAPFG, TAPEH, TAPEI, TAPFJ, TAPEK
C
C EQUIVALENCE (FLOW(1,1), B(1,1))
C
C DATA RC, NX, MX, MX1, KD
1 /0.1745329E-1, 503, 7, 1, 2000/
C GET CHECKOUT PRINT FLAG
I PRINT = 1
C
C INITIALIZE CONSTANT ARRAY FLOWC
DO 98 I = 1, 7
9A FLOWC(I) = 0.0
C
C IC = 0
NR = 0
C
C CYCLE ON SUB-REGIONS
```

SFNT 001
SFNT 002
SFNT 003
SFNT 004
SFNT 005
SFNT 006
SFNT 007
SFNT 008
SFNT 009
SFNT 010
SFNT 011
SFNT 012
SFNT 013
SFNT 014
SFNT 015
SFNT 016
SFNT 017
SFNT 018
SFNT 019
SFNT 020
SFNT 021
SFNT 022
SFNT 023
SFNT 024
SFNT 025
SFNT 026
SFNT 027
SFNT 028
SFNT 029
SFNT 030
SFNT 031
SFNT 032
SFNT 033
SFNT 034
SFNT 035

SFNTRP

SFNTRP

```

DO 600 II = 1, NS
ISRR = ISR(II)
IG10 = LOGC(IR) + 4 + ISBR
CALL READMS(10, E1, 25, IG10)
READ(10, ITAG10) E1
LOFF = E1(1) + 0.01
N1 = E1(6) + 0.01
C

DO 500 J = 1, 5
LOSF(J) = E1(J+20) + 0.01
IF (ISFR .EQ. 0) GO TO 510
C
IF (LOSF(ISFR) .LE. 0) GO TO 600
C
C READ IN SECONDARY FLOW POINTERS
IG10 = LOSF(ISFR)
CALL READMS(10, E1, 25, IG10)
READ(10, ITAG10) E1
LOFF = E1(1) + 0.01
N1 = E1(6) + 0.01
C
C READ IN THE DATA
510 ITAG10 = LOFF
ISRF = 0
IB = 0
DO 310 I = 1, N1
CALL READMS(10, E4, 12, ITAG10)
READ(10, ITAG10) E4
ITAG10 = ITAG10 + 1
C
C TRANSFORM DATA
301 CONTINUE
XX = E4(1) - XO
YY = E4(2) - YO
ZZ = E4(3) - ZO

```

```

SFNT 036
SFNT 037
SFNT 038
SFNT 039C
SFNT 040I
SFNT 041
SFNT 042
SFNT 043
SFNT 044
SFNT 045
SFNT 046
SFNT 047
SFNT 048
SFNT 049
SFNT 050
SFNT 051
SFNT 052C
SFNT 053I
SFNT 054
SFNT 055
SFNT 056
SFNT 057
SFNT 058
SFNT 059
SFNT 060
SFNT 061
SFNT 062C
SFNT 063I
SFNT 064
SFNT 065
SFNT 066
SFNT 067
SFNT 068
SFNT 069
SFNT 070

```

SFNTRP

SFNTRP

0FNT	071
8FNT	072
8FNT	073
8FNT	074
8FNT	075
8FNT	076
8FNT	077
8FNT	078
8FNT	079
8FNT	080
8FNT	081
8FNT	082
8FNT	083
8FNT	084
8FNT	085
8FNT	086
8FNT	087
8FNT	088
8FNT	089
8FNT	090
8FNT	091
8FNT	092
8FNT	093
8FNT	094
8FNT	095
8FNT	096
8FNT	097
8FNT	098
8FNT	099
8FNT	100
8FNT	101
8FNT	102
8FNT	103
8FNT	104
8FNT	105

SFNTRP

```

C
  XT(1) = XX*AP11 + YY*AP12 + ZZ*AP13
  XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
  XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
  XT(4) = XT(1)
  XT(5) = SQRT(XT(2)**2 + XT(3)**2)
  IF (XT(2) .EQ. 0.0) GO TO 304
  XT(6) = ATAN2(-T(2),XT(3))
  GO TO 305
304 XT(6) = 0.0
  IF (XT(3) .GT. 0.0) XT(6) = 3.141592654
305 CONTINUE
  IF (IBSF .EQ. 1) GO TO 523
C
C   NORMALIZE THE DATA
  IC = IC + 1
  IF (IC .LE. 500) GO TO 303
  WRITE (TAPE0,302)
302  FORMAT (1H0,4H*ROUTINE SFNTRP HAS ATTEMPTED TO LOAD MORE DATA ,
  2 ,1H ,52H*CALCULATIONS WILL CONTINUE WITH ONLY THE FIRST 100 ,
  3 7HPOINTS.
  )
  IC = IC + 1
  GO TO 311
303 YI(IC) = (XT(IY) - V1)/VL
  IF (ISURF .EQ. 1) GO TO 306
  XI(IC) = (XT(IX) - U1)/UL
  GO TO 307
306 CHX = CR + (CT*CR)*YI(IC)
  XLE = U1 + (UL*U1)*YI(IC)
  XI(IC) = (XT(IX) - XLE)/CHX
C
307 CONTINUE
C  CHECK IF WITHIN SURFACE BOUNDARIES.
  IF (XI(IC) .LT. 0.0) GO TO 308

```

SFNT-P

SFNT 106
 SFNT 107
 SFNT 108
 SFNT 109
 SFNT 110
 SFNT 111
 SFNT 112
 SFNT 113
 SFNT 114
 SFNT 115
 SFNT 116
 SFNT 117
 SFNT 118
 SFNT 119
 SFNT 120
 SFNT 121
 SFNT 122
 SFNT 123
 SFNT 124
 SFNT 125C
 SFNT 126I
 SFNT 127
 SFNT 128C
 SFNT 129I
 SFNT 130
 SFNT 131
 SFNT 132
 SFNT 133
 SFNT 134
 SFNT 135
 SFNT 136
 SFNT 137
 SFNT 138
 SFNT 139
 SFNT 140

```

    IF (XI(IC) .GT. 1.0) GO TO 308
    IF (YI(IC) .LT. 0.0) GO TO 30A
    IF (VI(IC) .GT. 1.0) GO TO 30B

C POINT WITHIN SURFACE BOUNDARIES. SET UP FLOW BOUNDARIES --
C -- INITIALIZE AS SURFACE BOUNDARIES.
    IF (IB .EQ. 1) GO TO 535
    IB = 1
    NB = NB + 1
    BX(1,NB) = 0.0
    BY(1,NB) = YI(IC)
    IF (ISFR .EQ. 0) GO TO 520
    BY(1,NB) = XI(IC)

C 520 CONTINUE
    IF (IFTYP .EQ. 2) GO TO 530
    IF (ISFR .EQ. 5) GO TO 530
    IG10SF = LOSF(ISFR+1)
    IF (IG10SF .LE. 0) GO TO 530
    CALL READMS(10,E1,E125,IG10SF)
    READ(10,IG10SF) E1
    LOFF = E1(1) + 0.01
    CALL READMS(10,E4B,12,LOFF)
    READ(10,LOFF) E4B
    E4(1) = E4B(1)
    E4(2) = E4B(2)
    E4(3) = E4B(3)
    IBSF = 1
    GO TO 301

C 523 IBSF = 0
    YIB = (XT(IY) - V1)/VL
    IF (ISURF .EQ. 1) GO TO 526
    XIA = (XT(IX) - U1)/UL
    GO TO 527

C 526 CHA = CH + (CTOCR)*YIB
  
```

SFNTRP

SFNTTRP

SFNT 141
 SFNT 142
 SFNT 143
 SFNT 144
 SFNT 145
 SFNT 146
 SFNT 147
 SFNT 148
 SFNT 149
 SFNT 150
 SFNT 151
 SFNT 152
 SFNT 153
 SFNT 154
 SFNT 155
 SFNT 156
 SFNT 157
 SFNT 158
 SFNT 159
 SFNT 160
 SFNT 161
 SFNT 162
 SFNT 163
 SFNT 164
 SFNT 165
 SFNT 166
 SFNT 167
 SFNT 168
 SFNT 169
 SFNT 170
 SFNT 171
 SFNT 172
 SFNT 173
 SFNT 174
 SFNT 175

XLB = U1 + (UL-U1)*YIB
 XIB = (XT(IX) - XLB)/CHB

C

527 CONTINUE
 BX(2,NB) = XIB
 BY(2,NB) = YIB
 GO TO 535

C

C

C

530 CONTINUE
 BX(2,NB) = 1.0
 BY(2,NB) = BY(1,NB)

C

POINT WITHIN BOUNDARIES. SET UP FLOW ARRAYS,
 535 CONTINUE

C

C

TRANSFORM FLOW DIRECTION COSINES

VT(1) = E4(8)*AP11 + E4(9)*AP12 + E4(10)*AP13
 VT(2) = E4(8)*AP21 + E4(9)*AP22 + E4(10)*AP23
 VT(3) = E4(8)*AP31 + E4(9)*AP32 + E4(10)*AP33
 SINPH = SIN(XT(6))
 COSPH = COS(XT(6))
 VT(4) = VT(1)
 VT(5) = VT(2)*SINPH + VT(3)*COSPH
 VT(6) = VT(2)*COSPH + VT(3)*SINPH
 FLOW(IC,1) = E4(7)
 FLOW(IC,2) = VT(IX)
 FLOW(IC,3) = VT(IY)
 FLOW(IC,4) = VT(IZ)
 FLOW(IC,5) = E4(11)
 FLOW(IC,6) = E4(12)
 FLOW(IC,7) = XT(IZ)
 GO TO 310

C

POINT NOT WITHIN BOUNDARIES. RESET COUNTER AND GO TO NEXT POINT.

SFNTTRP

SFNTTRP

```
308 IC = IC + 1
310 CONTINUE
600 CONTINUE
  IF (IC .GT. 2) GO TO 311
  WRITE(TAPEOT,1000) IC
1000 FORMAT(1H1, 5HONLY ,I3,29H POINTS WERE FOUND IN SFNTRP ,
1 56HTO BE WITHIN SURFACE BOUNDARIES. THIS IS INSUFFICIENT /
2 55H TO USE THE SURFACE SPLINE. RUN CONTINUES, BUT USER IS,
3 3AH CAUTIONED TO CHECK RESULTS CAREFULLY.)
  WRITE(TAPEOT,1005) NDSET,TABSET,IR,ISR,ISFR
1005 FORMAT(1H0,40HTHIS APPLIES TO THE FOLLOWING FLOW DATA /
1 12H DATA SET = ,I3, 10X17HALPHA=BETA SET = ,I3,
2 10X14HFLOW REGION = ,I3/
3 15H SUB-REGIONS = , 20I3/
4 18H SECONDARY FLOW = ,I3)
  ERRPR = 1
  RETURN
C
C ALL THE SURFACE DATA ARE IN PROPER ARRAYS.
C
C 311 N2 = IC
C
C CHECK BOUNDARY POINTS
  IF (NB .GT. 1) GO TO 700
  NB = 2
  BX(1,2) = BX(1,1)
  BX(2,2) = BX(2,1)
  BY(1,2) = 1.0
  BY(2,2) = 1.0
C
C NORMALIZE TO FLOW BOUNDARIES (IN XI ONLY).
C FIRST ARRANGE BOUNDARY DATA IN ORDER OF ASCENDING YI VALUES
700 IB = 0
710 IB = IB + 1
```

SFNT 176
SFNT 177
SFNT 178
SFNT 179
SFNT 180
SFNT 181
SFNT 182
SFNT 183
SFNT 184
SFNT 185
SFNT 186
SFNT 187
SFNT 188
SFNT 189
SFNT 190
SFNT 191
SFNT 192
SFNT 193
SFNT 194
SFNT 195
SFNT 196
SFNT 197
SFNT 198
SFNT 199
SFNT 200
SFNT 201
SFNT 202
SFNT 203
SFNT 204
SFNT 205
SFNT 206
SFNT 207
SFNT 208
SFNT 209
SFNT 210

SFNTTRP

SFNTRP

```

      I1 = 1
      720 VY = BY(IB,I1)
          I2 = I1 + 1
          J = C
          DO 730 I = I2,NB
          IF (VY,LT, BY(IB,I)) GO TO 730
          J = I
          VY = BY(IB,I)
      730 CONTINUE
C
      IF (J.EQ, 0) GO TO 740
      RY(IB,J) = BY(IB,I1)
      BY(IB,I1) = VY
C
      740 I1 = I1 + 1
          IF (I1.LT, NB) GO TO 720
          IF (IB.EQ, 1) GO TO 710
C
C   BOUNDARY DATA NOW IN ORDER, PROCEED WITH NORMALIZATION.
      IF (IPRINT.EQ, 0) GO TO 770
      WRITE(TAPEOT,750) NB
      750 FORMAT(1H1,1HBOUNDARY DATA, 13, 4H POINTS, /
          1 1H0, 3X1H1, 7X2HX1, 12X2HY1, 12X2HX2, 12X2HY2//)
          WRITE(TAPEOT,760) (I,BX(1,I),BY(1,I),BX(2,I),BY(2,I), I=1,NB)
      760 FORMAT(1H 13, 4F14.6)
      770 CONTINUE
      IC = 0
      DO 800 I = 1,N2
          XX = XI(I)
          VY = YI(I)
          IB = 0
      800 IB = IB + 1
C
      DO 810 I1 = 1,NB
          J = I1

```

SFNTRP

SFNTRP

```

      I1 = 1
      720 YY = BY(IB,I1)
      I2 = I1 + 1
      J = 0
      DO 730 I = I2,NB
      IF (YY .LT. BY(IR,I)) GO TO 730
      J = I
      YY = BY(IB,I)
      730 CONTINUE
C
      IF (J .EQ. 0) GO TO 740
      BY(IR,J) = BY(IB,I1)
      BY(IB,I1) = YY
C
      740 I1 = I1 + 1
      IF (I1 .LT. NB) GO TO 720
      IF (IB .EQ. 1) GO TO 710
C
C   BOUNDARY DATA NOW IN ORDER.  PROCEED WITH NORMALIZATION.
      IF (IPRINT .EQ. 0) GO TO 770
      WRITE(TAPEOT,750) NB
      750 FORMAT(1H1,1RHBOUNDARY DATA, 13, AH POINTS,/,
      1 1H0, 3X1H1, 7X2HX1, 12X2HY1, 12X2HX2, 12X2HY2//)
      WRITE(TAPEOT,760) (1,BX(1,I),BY(1,I),BX(2,I),BY(2,I), I=1,NB)
      760 FORMAT(1H , 13, 4F14.6)
      770 CONTINUE
      IC = 0
      DO 800 I = 1,N2
      XX = XI(I)
      YY = YI(I)
      IB = 0
      800 IB = IB + 1
C
      DO 810 I1 = 1,NB
      J = I1

```

8FNT 211
SFNT 212
8FNT 213
SFNT 214
8FNT 215
SFNT 216
8FNT 217
SFNT 218
8FNT 219
8FNT 220
8FNT 221
8FNT 222
8FNT 223
8FNT 224
8FNT 225
8FNT 226
8FNT 227
8FNT 228
8FNT 229
8FNT 230
8FNT 231
8FNT 232
8FNT 233
8FNT 234
8FNT 235
8FNT 236
8FNT 237
8FNT 238
8FNT 239
8FNT 240
8FNT 241
8FNT 242
8FNT 243
8FNT 244
8FNT 245

SFNTRP

SFNTRP

SFNT 246
SFNT 247
SFNT 248
SFNT 249
SFNT 250
SFNT 251
SFNT 252
SFNT 253
SFNT 254
SFNT 255
SFNT 256
SFNT 257
SFNT 258
SFNT 259
SFNT 260
SFNT 261
SFNT 262
SFNT 263
SFNT 264
SFNT 265
SFNT 266
SFNT 267
SFNT 268
SFNT 269
SFNT 270
SFNT 271
SFNT 272
SFNT 273
SFNT 274
SFNT 275
SFNT 276
SFNT 277
SFNT 278
SFNT 279
SFNT 280

```
      IF (VY .LT. BY(IB,I1)) GO TO A20
A10 CONTINUE
C
  820 IF (J .EQ. 1) J = 2
      DY = BY(IB,J) - BY(IB,J=1)
      IF (DY .EQ. 0.0) GO TO 900
      DX = BX(IB,J) - BX(IB,J=1)
      BP(IB) = BX(IB,J=1) + (VY - BY(IB,J=1))*DX/DY
      GO TO (A30,A40), IA
A30 IF (XX .LT. XP(IB)) GO TO 900
      GO TO 800
A40 IF (XX .GT. BP(IA)) GO TO 900
C
C POINT WITHIN FLOW BOUNDARIES
  IC = IC + 1
  XI(IC) = XX - BP(1)
  YI(IC) = YY
  IF (IVIS .EQ. 0) GO TO 850
  XI(IC) = XI(IC)/(BX(2,1) - BX(1,1))
  GO TO 860
A50 XI(IC) = XI(IC)/(BP(2) - BP(1))
A60 FLOW(IC,1) = FLOW(I,1)
      FLOW(IC,2) = FLOW(I,2)
      FLOW(IC,3) = FLOW(I,3)
      FLOW(IC,4) = FLOW(I,4)
      FLOW(IC,5) = FLOW(I,5)
      FLOW(IC,6) = FLOW(I,6)
      FLOW(IC,7) = FLOW(I,7)
C
      DO 300 II = 1,7
300 FLOWC(II) = FLOWC(II) + FLOW(IC,II)
C
  900 CONTINUE
      IF (IC .GT. 2) GO TO 910
      WRITE(TAPEOT,1010) IC
```

SFNTRP

SFNTRP

```

1010 FORMAT(1M1, 5HONLY ,I3, 20H POINTS WERE FOUND IN SFNTRP,
151MTO BE WITHIN FLOW BOUNDARIES, THIS IS INSUFFICIENT/
2 55H TO USE THE SURFACE SPLINE. RUN CONTINUES, BUT USER IS,
3 36H CAUTIONED TO CHECK RESULTS CAREFULLY.)
WRITE(TAPEOT,1005) N0SET,IARSET,IR,ISR,ISFR
ERROR = 1
RETURN
910 CONTINUE
N2 = IC
N3 = N2 + 3
C PRINT NORMALIZED ARRAYS
IF (IPRINT .EQ. 0) GO TO 330
WRITE(TAPEOT,3060) N0SET, IABSFT,IR
3060 FORMAT(1M1,34HNORMALIZED SURFACE DATA, N0SET = ,I2, 5X,
1 9HIARSET = ,I2, 5X, 9HREGION = ,I2/1H0, I4,1M1, I13,2MXI,
2 I27,2HYI, I39,4HMACH, I55,2HMX, I69,PHMY, I83,2HMZ,
3 I95,4MP/PS, I109,4HT/TS, I125,2HZI//)
WRITE(TAPEOT,3070) (I,XI(I), YI(I), (FLOW(I,J),J=1,7), I=1,N2)
3070 FORMAT(1M , I3, 9F14.6)
C
C
C 330 CONTINUE
DO 350 J = 1,7
FLOWC(J) = FLOWC(J)/N2
DO 340 I = 1,N2
340 FLOW(I,J) = FLOW(I,J) - FLOWC(J)
350 CONTINUE
C
C
C CALCULATE COEFFICIENT ARRAY (SURFACE SPLINE).
LS = 0
CALL ROWFM1 (N2,MX,XI,YI,FLOW,XKF,NX,MX,LS,TAPED)
CALL SOLVIT(AA,N3,MX,KD,TAPER,TAPEC,TAPED,B,NX,MX,NERR)

```

SFNT 281
SFNT 282
SFNT 283
SFNT 284
SFNT 285
SFNT 286
SFNT 287
SFNT 288
SFNT 289
SFNT 290
SFNT 291
SFNT 292
SFNT 293
SFNT 294
SFNT 295
SFNT 296
SFNT 297
SFNT 298
SFNT 299
SFNT 300
SFNT 301
SFNT 302
SFNT 303
SFNT 304
SFNT 305
SFNT 306
SFNT 307
SFNT 308
SFNT 309
SFNT 310
SFNT 311
SFNT 312
SFNT 313
SFNT 314
SFNT 315

SFNTRP

SFNTRP

```
IF (NERR, NE, 0) WRITE(TAPEOT,220)
220 FORMAT('HI, 16HMATRIX SINGULAR.')
C
C COEFFICIENTS ARE IN ARRAY B(I,J)
IF (IPRINT, FO, 0) GO TO 400
WRITE(TAPEOT,3080)
3080 FORMAT('HI, 42HSURFACE INTERPOLATION COEFFICIENTS B(I,J)')
3090 WRITE(TAPEOT,3090) (I, (B(I,J), J=1,7), I=1,N3)
FORMAT('H, IS, 7F15.5)
3100 WRITE(TAPEOT,3100) FLDWC
FORMAT('H0, 5X, 7F15.5)
400 CONTINUE
C
C
C
C
RETURN
END
```

SFNT 316
SFNT 317
SFNT 318
SFNT 319
SFNT 320
SFNT 321
SFNT 322
SFNT 323
SFNT 324
SFNT 325
SFNT 326
SFNT 327
SFNT 328
SFNT 329
SFNT 330
SFNT 331
SFNT 332
SFNT 333

SFNTRP

VALUE

```

SURROUTINE VALUE(N,X,Y,Z,XN,YN,ZN,IC,IT,INT,EMT,EMX,EMY,EMZ,POP,
1 TOT)
C
COMMON/SURFNT/INDRM,ISURF,IX,IY,IZ,U1,U2,V1,V2,CR,CT,CHX,N2,N3,
1 XO,YO,ZO,AP11,AP12,AP13,AP21,AP22,AP23,AP31,AP32,AP33,
2 FLOWC(7),XT(500),YI(500),B(503,7),XKF(503),A1(14400),
A X5(4),YB(4),ZB(4),
3 N0SET,IABSFT,IR,LORG(20),ISR(20),MS,TSFR,IVIS,IFTYP,IFLC,
4 BX(2,20),BY(2,20),NB
C
DIMENSION X(N),Y(N),Z(N),INT(N),EHT(N),EMX(N),EMY(N),EMZ(N),
1 POP(N),TOT(N),VT(6),XT(6),F1(7),BP(2)
C
DO 500 I = 1,N
IF (IC.EQ. 1) GO TO 70
C
TRANSFORM COORDS
XX = X(I) - XO
YY = Y(I) - YO
ZZ = Z(I) - ZO
C
XT(1) = XX*AP11 + YY*AP12 + ZZ*AP13
XT(2) = XX*AP21 + YY*AP22 + ZZ*AP23
XT(3) = XX*AP31 + YY*AP32 + ZZ*AP33
XT(4) = XT(1)
XT(5) = SORT(XT(2)**2 + XT(3)**2)
IF (XT(2).EQ. 0.) GO TO 40
XT(6) = ATAN2(XT(2),XT(3))
GO TO 50
40 XT(6) = 0.0
50 IF (XT(3).GT. 0.0) XT(6) = 3.141592654
50 CONTINUE
C
NORMALIZE COORDS

```

VALUE

VALU 001
VALU 002
VALU 003
VALU 004
VALU 005
VALU 006
VALU 007
VALU 008
VALU 009
VALU 010
VALU 011
VALU 012
VALU 013
VALU 014
VALU 015
VALU 016
VALU 017
VALU 018
VALU 019
VALU 020
VALU 021
VALU 022
VALU 023
VALU 024
VALU 025
VALU 026
VALU 027
VALU 028
VALU 029
VALU 030
VALU 031
VALU 032
VALU 033
VALU 034
VALU 035

VALUE

```

YN = (XT/IY) = VI)/VL
IF (ISURF .EQ. 1) GO TO 60
XN = (XT/IX) = UI)/UL
GO TO 70
60 CHX = CR + (CT=CR)*YN
XLE = UI + (UL=UI)*YN
XN = (XT/IX) = XLE)/CHX
C
C CHECK IF WITHIN SURFACE BOUNDARIES
70 IF (XN .LT. 0.0) GO TO 80
IF (XN .GT. 1.0) GO TO 80
IF (YN .LT. 0.0) GO TO 80
IF (YN .GT. 1.0) GO TO 80
C
C CHECK IF WITHIN FLOW BOUNDARIES
IB = 0
R00 IB = IB + 1
DO 810 I1 = 1,NR
J = I1
IF (YN .LT. RY(IR,I1)) GO TO 820
810 CONTINUE
C
820 IF (J .EQ. 1) J = 2
DY = BY(IB,J) = BY(IR,J=1)
IF (DY .EQ. 0.0) GO TO 80
DX = BX(IB,J) = BX(IR,J=1)
RP(IR) = BX(IR,J=1) + (YN = RY(IR,J=1))*DX/DY
GO TO (830,840), IB
830 IF (XN .LT. RP(IR)) GO TO 80
GO TO 800
840 IF (XN .GT. RP(I,J)) GO TO 80
C
C POINT WITHIN FLOW BOUNDARIES
XN = XN = RP(1)
IF (IVIS .EQ. 0) GO TO 850

```

VALUE

VALU	036
VALU	037
VALU	038
VALU	039
VALU	040
VALU	041
VALU	042
VALU	043
VALU	044
VALU	045
VALU	046
VALU	047
VALU	048
VALU	049
VALU	050
VALU	051
VALU	052
VALU	053
VALU	054
VALU	055
VALU	056
VALU	057
VALU	058
VALU	059
VALU	060
VALU	061
VALU	062
VALU	063
VALU	064
VALU	065
VALU	066
VALU	067
VALU	068
VALU	069
VALU	070

VALUE

VALU	071
VALU	072
VALU	073
VALU	074
VALU	075
VALU	076
VALU	077
VALU	078
VALU	079
VALU	080
VALU	081
VALU	082
VALU	083
VALU	084
VALU	085
VALU	086
VALU	087
VALU	088
VALU	089
VALU	090
VALU	091
VALU	092
VALU	093
VALU	094
VALU	095
VALU	096
VALU	097
VALU	098
VALU	099
VALU	100
VALU	101
VALU	102
VALU	103
VALU	104
VALU	105

```

      XN = XN/BX(2,1) - BX(1,1)
      GO TO 860
      850 XN = XN/BP(2) - BP(1)
      860 CONTINUE
C     POINT WITHIN BOUNDARIES, CALCULATE VALUES.
      XKF(1) = 1.0
      XKF(2) = XN
      XKF(3) = YN
      J3 = 3
      DO 250 J = 1,N2
      J3 = J3 + 1
      UIJ = 0.0
      TR1 = 0.0
      TR2 = 0.0
      IF (XN.EQ. XI(J)) GO TO 210
      TR1 = (XN - XI(J))*2
      210 IF (YN.EQ. YI(J)) GO TO 220
      TR2 = (YN - YI(J))*2
      220 RBIJ = TR1 + TR2
      IF (RBIJ.EQ. 0.0) GO TO 230
      UIJ = RBIJ*ALOG(RBIJ)
      230 XKF(J3) = UIJ
      250 CONTINUE
C
      DO 300 K = 1,7
      FIK(K) = FLOWC(K)
      DO 290 J = 1,N3
      FIK(K) = FIK(K) + XKF(J)*B(J,K)
      290 CONTINUE
      300 CONTINUE
C     INT(I) = 1

```

VALUE

VALUE

VALU 106
 VALU 107
 VALU 108
 VALU 109
 VALU 110
 VALU 111
 VALU 112
 VALU 113
 VALU 114
 VALU 115
 VALU 116
 VALU 117
 VALU 118
 VALU 119
 VALU 120
 VALU 121
 VALU 122
 VALU 123
 VALU 124
 VALU 125
 VALU 126
 VALU 127
 VALU 128
 VALU 129
 VALU 130
 VALU 131
 VALU 132
 VALU 133
 VALU 134
 VALU 135
 VALU 136
 VALU 137
 VALU 138
 VALU 139
 VALU 140

VALUE

```

PNT(I) = FI(1)
FY = SQRT(FI(2)**2 + FI(3)**2 + FI(4)**2)
IF (FT .EQ. 0.0) FT = 1.0
FI(2) = FI(2)/FT
FI(3) = FI(3)/FT
FI(4) = FI(4)/FT
EMX(I) = FI(2)/CR
EMY(I) = FI(3)/VL
PDF(I) = FI(5)
TOT(I) = FI(6)
ZN = FI(7)

C
C TRANSFORM XN, YN, ZN COORDS BACK TO (X, Y, Z)
IF (I.VIS .EQ. 0) GO TO 310
XN = IN*(BX(2,1) - BX(1,1))
GO TO 320
310 XN = XN*(BP(2) - BP(1))
320 XN = XN + BP(1)
XT(IY) = YN+VL + V1
XT(IZ) = ZN
IF (ISUMF .EQ. 1) GO TO 75
XT(IX) = XN+UL + U1
GO TO 76
75 XT(IX) = XN+CHX + U1 + (UL-U1)*YN
76 CONTINUE
IF (INORM .LE. C) GO TO 79
GO TO (A1, A2, A3, A4), INORM

C
C INORM = 0 IX = 4, IY = 5, IZ = 6
79 XX = XT(IX)
SINPH = SIN(XT(IZ))
COSPH = COS(XT(IZ))
YY = XT(IY)*SINPH
ZZ = -XT(IY)*COSPH
VT(1) = FI(2)

```

VALUE

VALU 141
VALU 142
VALU 143
VALU 144
VALU 145
VALU 146
VALU 147
VALU 148
VALU 149
VALU 150
VALU 151
VALU 152
VALU 153
VALU 154
VALU 155
VALU 156
VALU 157
VALU 158
VALU 159
VALU 160
VALU 161
VALU 162
VALU 163
VALU 164
VALU 165
VALU 166
VALU 167
VALU 168
VALU 169
VALU 170
VALU 171
VALU 172
VALU 173
VALU 174
VALU 175

```
VT(2) = FI(3)*SINPH + FI(4)*COSPH  
VT(3) = FI(4)*SINPH - FI(3)*COSPH  
GO TO 90
```

```
C  
C INORM = 1 IX = 1, IY = 2, IZ = 3  
61 XX = XT(IX)  
YY = XT(IY)  
ZZ = XT(IZ)  
110 VT(1) = FI(2)  
VT(2) = FI(3)  
VT(3) = FI(4)  
GO TO 90
```

```
C  
C INORM = 2 IX = 1, IY = 3, IZ = 2  
82 XX = XT(IX)  
YY = XT(IZ)  
ZZ = XT(IY)  
111 VT(1) = FI(2)  
VT(2) = FI(4)  
VT(3) = FI(3)  
GO TO 90
```

```
C  
C INORM = 3 IX = 2, IY = 3, IZ = 1  
83 XX = XT(IZ)  
YY = XT(IX)  
ZZ = XT(IY)  
112 VT(1) = FI(4)  
VT(2) = FI(2)  
VT(3) = FI(3)  
GO TO 90
```

```
C  
C INORM = 4 IX = 4, IY = 5, IZ = 5  
84 XX = XT(IX)  
SINPH = SIN(XT(IY))  
COSPH = COS(XT(IY))
```

VALUE

VALUE

VALU	176
VALU	177
VALU	178
VALU	179
VALU	180
VALU	181
VALU	182
VALU	183
VALU	184
VALU	185
VALU	186
VALU	187
VALU	188
VALU	189
VALU	190
VALU	191
VALU	192
VALU	193
VALU	194
VALU	195
VALU	196
VALU	197
VALU	198
VALU	199
VALU	200
VALU	201
VALU	202
VALU	203

```

      YV = XY(IZ)*SINPH
      ZZ = XT(IZ)*COSPH
      EMY(I) = EMY(I)/ZN
113  VT(1) = FI(2)
      VT(2) = FI(4)*SINPH + FI(3)*COSPH
      VT(3) = -FI(4)*COSPH + FI(3)*SINPH
C
      GO IF (IC .EQ. 0) GO TO 100
      X(1) = XC + XX*AP11 + YY*AP21 + ZZ*AP31
      Y(1) = YD + XX*AP12 + YY*AP22 + ZZ*AP32
      Z(1) = ZD + XX*AP13 + YY*AP23 + ZZ*AP33
C
      CHECK COORDINATE SYSTEM FOR DIRECTION COSINES
100  IF (IY .EQ. 1) GO TO 500
120  EMX(I) = VT(1)*AP11 + VT(2)*AP21 + VT(3)*AP31
      EMY(I) = VT(1)*AP12 + VT(2)*AP22 + VT(3)*AP32
      EMZ(I) = VT(1)*AP13 + VT(2)*AP23 + VT(3)*AP33
      GO TO 500
C
C POINT NOT WITHIN BOUNDARIES. SET FLAG AND GO TO NEXT POINT.
80  INT(I) = 0
C
500  CONTINUE
C
      RETURN
      END

```

VALUE

GRAPH

GRAP 001C
GRAP 002C
GRAP 003I
GRAP 004I
GRAP 005C
GRAP 006

OVERLAY (MARK4, S, 0)
PROGRAM GRAPH
SUBROUTINE GRAPH
RETURN
CONTINUE
END

C
C

GRAPH

AUXILI

AUXI	001C
AUXI	002C
AUXI	003I
AUXI	004
AUXI	005
AUXI	006
AUXI	007
AUXI	008
AUXI	009
AUXI	010
AUXI	011
AUXI	012
AUXI	013
AUXI	014
AUXI	015
AUXI	016
AUXI	017
AUXI	018
AUXI	019
AUXI	020
AUXI	021
AUXI	022
AUXI	023
AUXI	024
AUXI	025
AUXI	026
AUXI	027
AUXI	028
AUXI	029
AUXI	030
AUXI	031
AUXI	032
AUXI	033
AUXI	034
AUXI	035

AUXILI

```

OVERLAY (MARK4, 4, 0)
PROGRAM AUXILI
SURROUTINE AUXILI

C THIS IS THE EXECUTIVE CONTROL FOR THE AUXILIARY ROUTINES
C
C COMMON /EXEC/ CASE, TITLE(15), PAGE, ERROR
C
C COMMON /TAPE/ TAPEIN,TAPEOT,TAPEA,TAPER,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEN,TAPEJ,TAPEK
C
C INTEGER TAPEIN,TAPEOT,TAPEA,TAPER,TAPEC,TAPED,TAPEE,TAPEF,
1 TAPEG,TAPEN,TAPEJ,TAPEK
2 ,ERROR, PAGE
C
C DIMENSION IAUX(10)
C
C READ AUXILIARY CONTROL CARD
C READ(TAPEIN,10) IAUX
C 10 FORMAT(10I1)
C
C DO 70 I = 1,10
C IF (IAUX(I) .EQ. 0) GO TO 80
C IAX = IAUX(I)
C GO TO (20,30,40,50), IAX
C
C DATA SET LOAD OPTION
C 20 CONTINUE
C GO TO 60
C
C DATA SFT READ AND PRINT OPTION
C 30 CONTINUE

```

AUXILI

AUXI 036
AUXI 037
AUXI 038
AUXI 039
AUXI 040
AUXI 041
AUXI 042
AUXI 043
AUXI 044
AUXI 045
AUXI 046
AUXI 047
AUXI 048
AUXI 049
AUXI 050
AUXI 051
AUXI 052
AUXI 053
AUXI 054
AUXI 055I
AUXI 056C
AUXI 057

```
      GO TO 60
C DATA EDITING AND MAINTENANCE OPTION
 40 CONTINUE
      GO TO 60
C GENERAL CUTTING PLANE OPTION
 50 CONTINUE
      CALL GENCUT
      CALL QUTD
C
C      60 IF (ERROR .NE. 0) GO TO 99
 70 CONTINUE
C
      80 GO TO 110
 90 WRITE(TAPEOT,100) ERROR
100 FORMAT(1HG,39H***) ERROR IN AUXILIARY PROGRAM, ERROR = ,I2)
      STOP
C 110 RETURN
 110 CONTINUE
      END
```

AUXILI

GENCUT

```

C SUPROUTINE GENCUT
C GENERALIZED CUTTING PLANE OF MERIDIAN TYPE
C -- THAT IS, CUTTING PLANE IS PARALLEL TO AXIS
C
C
C     INTEGER SYMFCT
C     INTEGER ERROR, PAGE
C     INTEGER
C     TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
C
C     REAL LR
C     INTEGER NREC(40,100)
C     DIMENSION MINT(40), XREC(6), XP(4)
C
C     DIMENSION ELEM(22), XN(4), YN(4), ZN(4), YP(4), ZP(4), PHI(37),
C     1 A(4), MP(4), NPI(10), MPI(10),
C     2 XI(10,2), YI(10,2), ZI(10,2), RI(10,2), XA(10,2)
C     3 ,TITL(1), COMPID(10), NT(10), NI(10)
C     4 ,YPA(40), E(25), EP(25), IPANL(10)
C
C     COMMON /TAGS/ ITAG4, ITAG9, ITAG10
C
C     EQUIVALENCE (ELEM(11), XN(11)), (ELEM(15), YN(11)),
C     1 (ELEM(19), ZN(11))
C
C     COMMON /EXEC/ CASE, CTITLE(15), PAGE, FRPDR
C     COMMON /TAPE/ TI, TO, TAPEA, IUO1, IUO2,TAPEB,TAPEF,TAPEF,
C     1 TAPEG,TAPEH,TAPEI,IAPEJ,TAPEK
C
C     DATA PI,RC/3.14159265,1.74532925E-2/, NIMX/100/
C
C     READ IN CONFIGURATION TITLE
C     READ(TI, 5) CTITLE
C     5 FORMAT(15A4)
C     CALL HEADER

```

GENC 001
GENC 002
GENC 003
GENC 004
GENC 005
GENC 006
GENC 007
GENC 008
GENC 009
GENC 010
GENC 011
GENC 012
GENC 013
GENC 014
GENC 015
GENC 016
GENC 017
GENC 018
GENC 019
GENC 020
GENC 021
GENC 022
GENC 023
GENC 024
GENC 025
GENC 026
GENC 027
GENC 028
GENC 029
GENC 030
GENC 031
GENC 032
GENC 033
GENC 034
GENC 035

GENCUT

GENCUT

GENC 036
GENC 037
GENC 038
GENC 039
GENC 040
GENC 041
GENC 042
GENC 043
GENC 044
GENC 045
GENC 046
GENC 047
GENC 048
GENC 049
GENC 050
GENC 051
GENC 052
GENC 053
GENC 054
GENC 055
GENC 056
GENC 057
GENC 058
GENC 059
GENC 060
GENC 061
GENC 062
GENC 063
GENC 064
GENC 065
GENC 066
GENC 067
GENC 068
GENC 069
GENC 070

```
WRITE(TO, 6)
6 FORMAT(1H0;40H**** GENERAL CUTTING PLANE PROGRAM ****)

C
C READ IN TITLE AND UNIT CONTROLS
  READ(TI,10)TITLE, IORG, INPHI, MPL, ISYM, IPUNT
10 FORMAT(A4, 25X11, 4X11, 3X12, 4X11, 9X11)
  REWIND I/O1
  REWIND IU02

C
  IF (IORG .NE. 0) GO TO 20
  XPO = 0.0
  YPO = 0.0
  ZPO = 0.0
  PSTO = 0.0
  TWFO = 0.0
  PHIO = 0.0
  GO TO 40

C
C READ IN CUTTING PLANE ORIGIN AND ORIENTATION
C
C
C
20 CONTINUE
  READ(TI,30) XPO,YPO,ZPO,PSIO,THEO,PHIO
30 FORMAT(6F10.0)
C
C CONSTANTS RELATING TO CUTTING PLANE AXIS
40 CONTINUE
  SIMPS = SIN(PC*PSIO)
  COSPS = COS(PC*PSIO)
  COST = COS(PC*THEO)
  SINT = SIN(PC*THEO)
  STSPS = SIMPS*SINT
  STCPS = COSPS*SINT
  CTSPS = COST*COSPS
  CTSPPS = COST*SINPS
```

GENCUT

GENCUT

```
C      JJ = 1
C      IF (NPL.GT. 36) NPL = 36
C      IF (INPHI.LT. 2) GO TO 130
C      PARALLEL CUTTING PLANES.  READ IN CONSTANT PHI VALUE.
C      READ(TI,50)PHICD
C      50 FORMAT(F10.6)
C      PHI(1) = RC*PHICD
C      TANP = TAN(PHI(1))
C      IF (INPHI.EQ. 3) GO TO 90
C      EQUAL SPACING,  READ IN END VALUES.
C      READ(TI,60) (XN(I),YN(I),ZN(I), I=1,P)
C      60 FORMAT(3F10.0)
C      FOR EQUAL SPACING, MUST HAVE AT LEAST TWO PLANES
C      IF (NPL.LT. 2) NPL = 2
C      PROJECT INTO VIEWING PLANE AND FIND AXIS LOCATIONS.
C      DO 70 N = 1,2
C      XX = XN(N) - XPD
C      YY = YN(N) - YPD
C      ZZ = ZN(N) - ZPD
C      YP(N) = -XX*SINPS + YY*COSPS
C      ZP(N) = XX*STCPS + YY*STSPS + ZZ*COST
C      70 YPA(N) = YP(N) + ZP(N)*TANP
C      DYPA = (YPA(2) - YPA(1))/(NPL - 1)
C      NINT(1) = 0
C      N.L1 = NPL
C      DO 80 N = 2,NPL1
C      PHI(N) = PHI(1)
```

GENC 071
GENC 072
GENC 073
GENC 074
GENC 075
GENC 076
GENC 077
GENC 078
GENC 079
GENC 080
GENC 081
GENC 082
GENC 083
GENC 084
GENC 085
GENC 086
GENC 087
GENC 088
GENC 089
GENC 090
GENC 091
GENC 092
GENC 093
GENC 094
GENC 095
GENC 096
GENC 097
GENC 098
GENC 099
GENC 100
GENC 101
GENC 102
GENC 103
GENC 104
GENC 105

GENCUT

GENCUT

GENC 106
GENC 107
GENC 108
GENC 109
GENC 110
GENC 111
GENC 112
GENC 113
GENC 114
GENC 115
GENC 116
GENC 117
GENC 118
GENC 119
GENC 120
GENC 121
GENC 122
GENC 123
GENC 124
GENC 125
GENC 126
GENC 127
GENC 128
GENC 129
GENC 130
GENC 131
GENC 132
GENC 133
GENC 134
GENC 135
GENC 136
GENC 137
GENC 138
GENC 139
GENC 140

```

      NINT(N) = 0
      R0 YPA(N) = YPA(N-1) + DYP
      GO TO 110

      C
      C VARIABLE PLANE SPACING.          USITIONS,
      C PROJECT INTO VIEWING PLANE AND AXIS LOCATIONS.
      C
      NPL1 = NPL
      DO 100 N = 1, NPL
      READ(TI,60) XN(1), YN(1), ZN(1)
      XY = XN(1) * XPD
      YY = YN(1) * YPD
      ZZ = ZN(1) * ZPD
      YP(1) = -XX*SINPS + YY*COSPS
      ZP(1) = XX*STCPS + YY*STSPS + ZZ*CCOST
      YPA(N) = YP(1) + ZP(1)*FANP
      PHI(N) = PHI(1)
      NINT(N) = 0
      100 CONTINUE
      110 IF (IPRNT .EQ. 1)
      1WRITE(10,120)NPL, PHICD, (YPA(N), N = 1,NPL)
      120 FORMAT(1H1, I3, 32H PARALLEL CUTTING PLANES PHI = , F10.6/
      1 1H0, (6F20.6))
      GO TO 180

      C
      C
      C
      130 CONTINUE
      IF (INPHI .EQ. 0) GO TO 160
      READ(TI,140) (PHI(I), I=1,NPL)
      140 FORMAT(6F10.6)
      DPHI = PHIO+RC
      DO 150 I = 1,NPL
      NINT(I) = 0
      YPA(I) = 0.0

```

GENCUT

GENCUT

```
150 PHI(I) = PHI(I)*RC + DPHI
    NPLI = NPL
    GO TO 190
160 CONTINUE
    NPLI = NPL + 1
    DPHI = 360./(NPL) *RC
    PHI(I) = PHIC*RC
    NINT(I) = 0
    YPA(I) = 0.0
    DO 170 I = 2,NP.1
        NINT(I) = 0
        YPA(I) = 0.0
170 PHI(I) = PHI(I-1) + DPHI
C
    PHI(NPLI) = PHI(1)
180 MPTOT = 0
    DO 190 I = 1,10
190 NT(I) = 0
C
C   RETRIEVE ELEMENT DATA FROM STORAGE
    IGA = 2
    CALL READMS (4,E,25,IGA)
    READ (4,IG4) E
    NPF = E(2)
    NPMAX = E(3)
    NREM = F(4)
C
C   START OF PANEL CYCLE
    DO 675 JJI,10
        NT(JJ) = MPTOT + 1
        READ (11,210) IPANL(JJ),IPRINT,LAST
210 FORMAT (14,5X.1,4X,11)
```

GENC 141
GENC 142
GENC 143
GENC 144
GENC 145
GENC 146
GENC 147
GENC 148
GENC 149
GENC 150
GENC 151
GENC 152
GENC 153
GENC 154
GENC 155
GENC 156
GENC 157
GENC 158
GENC 159
GENC 160
GENC 161
GENC 162
GENC 163
GENC 164C
GENC 165I
GENC 166
GENC 167
GENC 168
GENC 169
GENC 170
GENC 171
GENC 172
GENC 173
GENC 174
GENC 175

GENCUT

GENCUT

GENC 176
 GENC 177
 GENC 178
 GENC 179C
 GENC 180I
 GENC 181
 GENC 182
 GENC 183
 GENC 184
 GENC 185
 GENC 186
 GENC 187
 GENC 188
 GENC 189
 GENC 190
 GENC 191C
 GENC 192I
 GENC 193
 GENC 194
 GENC 195
 GENC 196
 GENC 197
 GENC 198
 GENC 199
 GENC 200
 GENC 201
 GENC 202
 GENC 203
 GENC 204
 GENC 205
 GENC 206
 GENC 207
 GENC 208
 GENC 209
 GENC 210

```

IF (IPANL(JJ) .EQ. 0) GO TO 600
IF (TPANL(JJ) .GT. NPE) GO TO 900
IG4 = IPANL(JJ)*5
CALL READMS (4,EP,25,IG4)
READ (4*IG4) EP
COMMID = EP(2)
ISTART = EP(3)
LL = EP(4)
SYMFACT = EP(6)
COMMID(JJ) = COMMID
ISTART = ISTART + NREM

C START OF ELEMENT DO LOOP
DO 610 J=1,LI
IG4 = ISTART + NREM*J
CALL READMS (4,ELEM,25,IG4)
READ (4*IG4) ELEM
JE = J

C PROJECT FOUR CORNER POINTS INTO 12,13 PLANE
C (IN COORDS RELATIVE TO CUTTING PLANE AXYS),
C FIRST CHECK IF DUMMY, ZERO AREA ELEMENT - IF SO, SKIP OVER,
C IF (ELEM(10) .LE. 1.0E-5) GO TO 670

C ALWAYS PROJECT LINES 2=3 AND 3=4
NITZ = 2
NR = 2
NF = 4
ICOL = ELEM(2) + 0.01
IRDW = ELEM(3) + 0.01
C CHECK IF NEW ROW
IF (IROW .NE. 1) GO TO 250
NR = 1
NF = 1
C CHECK IF NEW COLUMN
  
```

GENCUT

GENCUT

```
250 IF (ICOL .NE. 1) GO TO 260
    NII = 1
    NR = 1
260 CONTINUE
C
C PRESET NPI ARRAY
    DO 270 K = 1,10
270 NPI(K) = 0
C
    IO = 0
    DO 360 N = NR,4
    XX = XN(N) = XPO
    YY = YN(N) = YPO
    ZZ = ZN(N) = ZPO
C
    XP(N) = XX*CTSPS + YY*CTSPS + ZZ*SINT
    YP(N) = -XX*SINPS + YY*COSSPS
    ZP(N) = XX*STSPS + YY*STSPS + ZZ*CCOST
C
    MP(N) = 0
    IF (INPHI .GT. 1) GO TO 290
    IF ((ZP(N).EQ.0.0) .AND. (YP(N).EQ.0.0)) GO TO 350
    IF (JE .GT. LL) GO TO 280
    IF ((ZP(N).LT.0.0) .AND. (YP(N).EQ.0.0)) GO TO 360
280 CONTINUE
C
    A(N) = ATAN2(+YP(N),+ZP(N))
    IF (A(N) .LE. 0.0) A(N) = 2.*PI + A(N)
    IF (INPHI .NE. 0) GO TO 320
    MP(N) = A(N)/DPHI + 1
    GO TO 360
C
C MP CALCULATIONS FOR PARALLEL PLANES
290 CONTINUE
    A(N) = YP(N) + ZP(N)*TANP
```

GENC 211
GENC 212
GENC 213
GENC 214
GENC 215
GENC 216
GENC 217
GENC 218
GENC 219
GENC 220
GENC 221
GENC 222
GENC 223
GENC 224
GENC 225
GENC 226
GENC 227
GENC 228
GENC 229
GENC 230
GENC 231
GENC 232
GENC 233
GENC 234
GENC 235
GENC 236
GENC 237
GENC 238
GENC 239
GENC 240
GENC 241
GENC 242
GENC 243
GENC 244
GENC 245

GENCUT

GENCUT

```
IF (INPHI .EQ. 3) GO TO 300
PM = (A(N) - YPA(1))/DYPA
IF (PM .LE. 0.0) GO TO 360
MP(N) = PM + 1
IF (MP(N) .GT. NPL) MP(N) = N.L
GO TO 360

C
300 CONTINUE
DO 310 K = 1,NPL1
IP = K
IF (YPA(K) .GT. A(N)) GO TO 340
310 CONTINUE
MP(N) = IP
GO TO 360

C
C
320 DO 330 K = 1,NPL1
IP = K
IF (PHI(K) .GT. A(N)) GO TO 340
330 CONTINUE
IF (JE .GT. LL) GO TO 360
IP = IP + 1
340 MP(N) = IP - 1
GO TO 360
350 IO = IO + 1
360 CONTINUE

C
C CHECK FOR CONTINUITY IN ANGLE (REFLECTED SYMMETRIC ELEMENTS ONLY).
IF (JE .LE. LL) GO TO 380
DO 370 N = NR,d
IF (MP(N) .EQ. 1) MP(N) = NPL1
370 CONTINUE
380 CONTINUE

C
C CHECK FOR ORIGIN POINTS (ASSUMED IN PAIRS)
```

GENC 246
GENC 247
GENC 248
GENC 249
GENC 250
GENC 251
GENC 252
GENC 253
GENC 254
GENC 255
GENC 256
GENC 257
GENC 258
GENC 259
GENC 260
GENC 261
GENC 262
GENC 263
GENC 264
GENC 265
GENC 266
GENC 267
GENC 268
GENC 269
GENC 270
GENC 271
GENC 272
GENC 273
GENC 274
GENC 275
GENC 276
GENC 277
GENC 278
GENC 279
GENC 280

GENCUT

GENCUT

```
IF (J0 .EQ. 0) GO TO 420
N1 = 1
N2 = 2
N3 = 3
N4 = 4
390 IF (MP(N1) .NE. 0) GO TO 410
IF (MP(N2) .EQ. 0) GO TO 400
C
MP(N1) = MP(N2)
MP(N2) = MP(N3)
GO TO 420
C
400 MP(N1) = MP(N2)
MP(N2) = MP(N3)
GO TO 420
C
410 NN = N1
N1 = N2
N2 = N3
N3 = N4
N4 = NN
IF (N4 .NE. 4) GO TO 390
C
420 CONTINUE
C
C
C TEST FOR INTERSECTIONS AND FIND THEM IF INDICATED
MPIN = 1
NUMP = 1
ICT = +1
KPMAX = 0
IPI = NII
C
430 IP2 = IPI + 1
IF (IP2 .GT. 4) IP2 = 1
```

GENCUT

GENCUT

```
C      IF (MP(IP2) = MP(IP1))460,440,450
C
C      CHECK FOR LAST CORNER POINT
C      440 IF (IP2 .EQ. NF) GO TO 600
C
C      GO TO NEXT CORNER POINT
C      IP1 = I22
C      GO TO 430
C
C      INTERSECTION OF NMP = MP(IP1) + 1 WITH LYNE SEGMENT
C      450 NMP = MP(IP1) + 1
C      IMP = +1
C      GO TO 470
C
C      INTERSECTION OF NMP = MP(IP1) WITH LYNE SEGMENT
C      460 NMP = MP(IP1)
C      TMP = -1
C
C      CHECK FOR FIRST INTERSECTION
C      470 IF (MPMAX .EQ. 0) GO TO 540
C
C      PREVIOUS INTERSECTIONS, CHECK FOR TURNING CORNER
C      AND COMING BACK.
C      IF (IMP)480,480,500
C
C      480 IF (MP(IP1) = MP(IP1-1))530,490,520
C
C      490 IF (MP(IP1-1) = MP(IP1-2))530,520,520
C
C      500 IF (MP(IP1) = MP(IP1-1))520,510,530
C
```

GENC 314
GENC 317
GENC 318
GENC 319
GENC 320
GENC 321
GENC 322
GENC 323
GENC 324
GENC 325
GENC 326
GENC 327
GENC 328
GENC 329
GENC 330
GENC 331
GENC 332
GENC 333
GENC 334
GENC 335
GENC 336
GENC 337
GENC 338
GENC 339
GENC 340
GENC 341
GENC 342
GENC 343
GENC 344
GENC 345
GENC 346
GENC 347
GENC 348
GENC 349
GENC 350

GENCUT

GENCUT

```
C
C
510 IF (MP(IP1-1) * MP(IP1-2))520,520,530
C
C YES, REVERSING DIRECTION
520 ICT = -1
GO TO 550
C
530 NUMP = NUMP + ICT
IF (ICT .GT. 0) GO TO 540
C
C CHECK FOR INITIAL CUTTING PLANE
IF (NUMP .GE. MPIN) GO TO 550
C
C HAVE COME BACK ACROSS INITIAL PLANE
C CHANGE SIGN OF ICT AND INCREASE MPMAX
ICT = -1
NUMP = MPMAX + 1
MPIN = NUMP
C
540 IF (NUMP .GT. MPMAX) MPMAX = NUMP
C
C
C CALCULATE INTERSECTION
550 NPT(NUMP) = NPI(NUMP) + 1
NP = NPI(NUMP)
MPI(NUMP) = NPS
DY = YB(IP2) - YP(IP1)
DZ = ZP(IP2) - ZP(IP1)
CSP = COS(PI*PHI(NMP))
SNP = SIN(PI*PHI(NMP))
LR = 0.0
RI(NUMP, NP) = 0.0
OYDZ = DY**2 + DZ**2
IF (OYDZ .LE. 1.0E-6) GO TO 560
C
```

```
GENC 351
GENC 352
GENC 353
GENC 354
GENC 355
GENC 356
GENC 357
GENC 358
GENC 359
GENC 360
GENC 361
GENC 362
GENC 363
GENC 364
GENC 365
GENC 366
GENC 367
GENC 368
GENC 369
GENC 370
GENC 371
GENC 372
GENC 373
GENC 374
GENC 375
GENC 376
GENC 377
GENC 378
GENC 379
GENC 380
GENC 381
GENC 382
GENC 383
GENC 384
GENC 385
```

GENCUT

GENC 386
 GENC 387
 GENC 388
 GENC 389
 GENC 390
 GENC 391
 GENC 392
 GENC 393
 GENC 394
 GENC 395
 GENC 396
 GENC 397
 GENC 398
 GENC 399
 GENC 400
 GENC 401
 GENC 402
 GENC 403
 GENC 404
 GENC 405
 GENC 406
 GENC 407
 GENC 408
 GENC 409
 GENC 410
 GENC 411
 GENC 412
 GENC 413
 GENC 414
 GENC 415
 GENC 416
 GENC 417
 GENC 418
 GENC 419
 GENC 420

```

DYP = YP(IP1) - YPA(NMP)
DZP = ZP(IP1)
RIMAG = (DZP*DY + DYP*DZ)/(CSP*DY + SNP*DZ)
DPROD = -(ELEM(5)*COSPS - ELEM(4)*SINPS)*SNP
      + (ELEM(4)*STCPS + ELEM(5)*STSPS + ELEM(6)*COST)*CSP
RI(NUMP, NP) = RIMAG
IF (IAPHI, LT, 2)
  IRI(NUMP, NP) = SIGN(RIMAG, DPROD)
  YPI = SNP*RI(NUMP, NP) + YPA(NMP)
  ZPI = CSP*RI(NUMP, NP)
  LR = SQRT(((YPI-YP(IP1))**2 + (ZPI-ZP(IP1))**2)/DYDZ)
  RI(NUMP, NP) = ARS(RIMAG)
C 560 CONTINUE
  XA(NUMP, NP) = XE(IP1) + (XP(IP2) - XE(IP1))*LR
  XI(NUMP, NP) = XN(IP1) + (XN(IP2) - XN(IP1))*LR
  YI(NUMP, NP) = YN(IP1) + (YN(IP2) - YN(IP1))*LR
  ZI(NUMP, NP) = ZN(IP1) + (ZN(IP2) - ZN(IP1))*LR
  IF (NINT(NMP) .LT. NIMX) GO TO 566
  WRITE(10, 565) NMP, NIMX
565 FORMAT(1H0,45H THE NUMBER OF INTERSECTIONS ON CUTTING PLANE, I3,
1 13H EXCEEDS THE MAXIMUM ALLOWABLE (,I3, 2H)./1H0,
2 54H RUN CONTINUES BUT FURTHER INTERSECTIONS ARE IGNORED.)
  GO TO 570
C 566 CONTINUE
  NINT(NMP) = NINT(NMP) + 1
  RJ = JF
  WRITE(IUO1)RJ, XI(NUMP, NP), YI(NUMP, NP), ZI(NUMP, NP), RI(NUMP, NP)
1 , XA(NUMP, NP)
  MPTOT = MPTOT + 1
  NRFC(NMP, NINT(NMP)) = MPTOT
C
C
C CONTINUE TO NEXT INTERSECTION

```

GENCUT

```
570 NMP = NMP + IMP
    IF (TMP)580,580,590
580 IF (MP(IP2) = NMP)530,440,440
590 IF (MP(IP2) = NMP)440,530,530
C
C
C ALL INTERSECTIONS FOUND ON CURRENT ELEMENT. SAVE THE RESULTS.
600 CONTINUE
    IF (MPMAX .EQ. 0) GO TO 650
    IF (IPRINT .FO. 0) GO TO 650
    WRITE(TO,610)
610 FORMAT(1H0, 4H ID, 714, 3HNO., T24, 2HMP, T45, 1HX, T65, 1HY,
1 T85, 1HZ, T105, 1HR, T125, 1HA/)
C
    DO 620 K = 1,MPMAX
    NPIK = NPI(K)
620 WR... (TO,630) COMPID(JJ), JE, MPI(K),
1 XI(K,IP), YI(K,IP), ZI(K,IP), RI(K,IP),
2 XA(K,IP), IP = 1,NPIK)
630 FORMAT(1H , 1XA4, 6X13, 7X13, 4X, 5F20.6/1H , 28X5F20.6)
C
C
C RIGHT OUT ELEMENT CORNER POINTS
WRITE(TO,640) (I, XN(I), YN(I), ZN(I), I=1,4)
640 FORMAT(1H , 21X13, 4X, 3F20.6)
C
C
C CHECK FOR SYMMETRY
650 IF (SYMFACT .NE. 0) GO TO 670
    IF (ISYM .EQ. 0) GO TO 670
    IF (JE .GT. LL) GO TO 670
    DO 660 N = NF,4
660 YN(N) = -YN(N)
```

GENCUT

GENCUT

```

JE = LL + J
GO TO 260

C
C
670 CONTINUE
IF (LAST .NE. 0) GO TO 680
675 CONTINUE
C END OF PANEL DO LOOP
C
C
C ARRANGE INTERSECTIONS AS PER MERIDIAN
C FIRST CHECK IF ANY INTERSECTIONS FOUND
680 IF (MPTOT .GT. 0) GO TO 700
WRITE(TD,690) TITLE
690 FORMAT(:M0,4)NO INTERSECTIONS FOUND FOR CONFIGURATION ,
1 A4/1MC, 16HCASE TERMINATED,)
STOP
C
700 CONTINUE
NF = NINT(I)
IF (YNPHI .EQ. 0) NINT(I) = NINT(NPL1)
WRITE(IU02) TITLE, COMPID, JJ, NT, NPL, MPTOT,
: PSID, TME0, PHIO, XPO, YPO, ZPO, YNPHI,
2 (PHI(I), I=1,NPL), (NINT(I), I=1,NPL), (YPA(I), I=1,NPL)
NINT(1) = NF
DO 830 I = 1, NPL
II = I
710 NF = NINT(II)
IF (NF .EQ. 0) GO TO A20
K = 0
K1 = 0
K2 = 1
K3 = 0
REWIND IU01

```

GENC 456
GENC 457
GENC 458
GENC 459
GENC 460
GENC 461
GENC 462
GENC 463
GENC 464
GENC 465
GENC 466
GENC 467
GENC 468
GENC 469
GENC 470
GENC 471
GENC 472
GENC 473
GENC 474
GENC 475
GENC 476
GENC 477
GENC 478
GENC 479
GENC 480
GENC 481
GENC 482
GENC 483
GENC 484
GENC 485
GENC 486
GENC 487
GENC 488
GENC 489
GENC 490

GENCUT

GENCUT

```
C ZERO OUT COMPONENT COUNTERS
  00 720 L = 1,JJ
  720 NI(L) = 0
C
  00 810 J = 1,NF
  NN = NREC(II,J)
  730 K = K + 1
  READ(IUOI,END=830) XREC
  READ (IUOI) XREC
  IF (EOF(IUOI)) 830,731
  731 IF (K.NE. NN) GO TO 730
  IF (K3 .EQ. 1) GO TO 780
C
C CHECK ON COMPONENT
  740 K1 = K1 + 1
  IF (K1 .GT. JJ) GO TO 810
  NI(K1) = 0
  NT1 = NT(K1)
  IF (NT1 .LE. 0) GO TO 740
C
  750 K2 = K2 + 1
  IF (K2 .EQ. K1) GO TO 750
  IF (K2 .LE. JJ) GO TO 760
  NTP = MPTOT + 1
  GO TO 770
  760 NT2 = NT(K2)
  IF (NT2 .EQ. 0) GO TO 750
  770 K3 = 0
C
C TEST IF RECORD WITHIN RANGE
  780 IF ((NN,GE,NT1).AND.(NN,LT,NT2)) GO TO 790
C
C NOT IN RANGE = GO TO NEXT COMPONENT
  GO TO 740
C
```

```
GENC 491
GENC 492
GENC 493
GENC 494
GENC 495
GENC 496
GENC 497
GENC 498I
GENC 499C
  ENC 500C
GENC 501
GENC 502
GENC 503
GENC 504
GENC 505
GENC 506
GENC 507
GENC 508
GENC 509
GENC 510
GENC 511
GENC 512
GENC 513
GENC 514
GENC 515
GENC 516
GENC 517
GENC 518
GENC 519
GENC 520
GENC 521
GENC 522
GENC 523
GENC 524
GENC 525
```

GENCUT

GENCUT

```
790 YF (K3 ,EQ, 1) GO TO A00
NICK1) = J
K3 = 1
C
A00 WRITE(IU02) K1, XREC
A10 CONTINUE
C
C ALL ELEMENTS IN CURRENT MERIDIAN STORED.
C SAVE COMPONENTS COUNTERS
WRITE(IU02) (NI(L), L=1,JJ)
C
820 IF (II ,NE, 1) GO TO A30
IF (INPHT ,GT, 1) GO TO B30
IF (NF ,NE, 0) GO TO A30
II = NPL1
GO TO 710
A30 CONTINUE
C
RETURN
C
900 WRITE (10,910)
910 FORMAT (1H ,43H**PANEL NUMBER IS GREATER THAN THE MAXIMUM ,
1 34HSTORED ON UNIT 4 ** PROGRAM STOP )
STOP
END
```

GENC 526
GENC 527
GENC 528
GENC 529
GENC 530
GENC 531
GENC 532
GENC 533
GENC 534
GENC 535
GENC 536
GENC 537
GENC 538
GENC 539
GENC 540
GENC 541
GENC 542
GENC 543
GENC 544
GENC 545
GENC 546
GENC 547
GENC 548
GENC 549
GENC 550

GENCUT

OUTD

001 OUTD
 002 OUTD
 003 OUTD
 004 OUTD
 005 OUTD
 006 OUTD
 007 OUTD
 008 OUTD
 009 OUTD
 010 OUTD
 011 OUTD
 012 OUTD
 013 OUTD
 014 OUTD
 015 OUTD
 016 OUTD
 017 OUTD
 018 OUTD
 019 OUTD
 020 OUTD
 021 OUTD
 022 OUTD
 023 OUTD
 024 OUTD
 025 OUTD
 026 OUTD
 027 OUTD
 028 OUTD
 029 OUTD
 030 OUTD
 031 OUTD
 032 OUTD
 033 OUTD
 034 OUTD
 035 OUTD

```

SURROUTINE OUTD
DIMENSION PHI(40), NINT(40), COMPID(10), NI(10), TITLE(1), NE(10),
1 RJ(100),X(100),Y(100),Z(100),R(100),A(100),YPA(40), AX(100),
2 J(100),JX(100),INR(10), IO(100),DORD(4)
COMMON /EXEC/ CASE,CTITLE(15), PAGE, ERROR
COMMON /TAPE/ TI, TO, TAPEA,TAPEB, IUIN,TAPEJ,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
INTEGER TI, TO, TAPEA,TAPEB, IUIN,TAPEJ,TAPEE,TAPEF,
1 TAPEG,TAPEH,TAPEI,TAPEJ,TAPEK
INTEGER PAGE, ERROR
DATA RC/1,74532925E-2/
DATA DORD/1HA, 1MX, 1HY, 1HZ/

C READ IN CONTROL CARD
C READ(TI,10) IOR, IOG, COMPR, IOY, IPRINT
C IO FORMAT(10I1, 3X12, 5XA4, 5X11, 24X11)

ON 20 I = 1,40
PHI(I) = 0.0
YPA(I) = 0.0
20 CONTINUE
REWIND IUIN
IT2 = 0

C IOR(I) = FLAG TO ORDER INDIVIDUAL COMPONENT (RANDOM)
C IOG = FLAG TO ORDER COMPONENTS BY GROUPS
C COMPR = BASE COMPONENT (OTHERS ADDED TO THIS ONE)
C POINTS FOR A PARTICULAR MERIDIAN ARE BROUGHT INTO CORE
C INDIVIDUAL COMPONENTS ARE THEN ORDERED (IF REQUIRED, NON-ZERO IOR)
C NEXT, GROUPS ARE ORDERED (IF REQUIRED, NON-ZERO, POSITIVE IOG)
C NOTE, A NEGATIVE IOG WILL CAUSE RANDOM ORDERING OF THE WHOLE SET.

C READ(IUIN) TITLE, COMPID, NC, NE, NPL, MPTOT,
C PSIO, TWFO, PHIO, XPO, YPO, ZPO, ITP1,

```

OUTD

OUTD

036
 037
 038
 039
 040
 041
 042
 043
 044
 045
 046
 047
 048
 049
 050
 051
 052
 053
 054
 055
 056
 057
 058
 059
 060
 061
 062
 063
 064
 065
 066
 067
 068
 069
 070

```

2 (PHI(I), I=1,NPL), (NINT(I), I=1,NPL), (YPA(I), I=1,NPL)
C
IF (IPRINT.EQ. 1)
1WRITE(70,30) TITLE, NPL, MPTOT,
2 PHID, PSIO, THEL, XPO, YPO, ZPO
30 FORMAT(1H1, T10, 15HCONFIGURATION, A4, T40,
1 17HNUMBER OF CUTS = , I2, T80, 15HTOTAL POINTS = , I4/1H0,
2 7HPHID = , F11.6, 5X, 7HPSIO = , F11.6, 5X,
3 7HTHEL = , F11.6, 5X, 5HXO = , F11.6, 5X,
4 5HYO = , F11.6, 5X, 5HZO = , F11.6/)
C
DO 500 KK= 1,NPL
C
C BRING MERIDIAN DATA INTO CORE
NF = NINT(KK)
NT = NF
PHID = PHI(KK)/RC
IF (IPRINT.EQ. 1)
1WRITE(70,40)KK,PHID, YPA(KK), MINT(KK)
40 FORMAT(1H0, 13HPLANE NUMBER , I2, T26, 6HPHI = , F11.6, 6X,
1 6HYPA = , F11.6,
2 T71, 26HNUMBER OF INTERSECTIONS = , I3/)
IF (NF.EQ. 0) GO TO 500
DO 50 I = 1,NF
J(I) = I
JX(I) = I
50 READ(IUIN)ID(I), RJ(I), X(I), Y(I), Z(I), R(I), AX(I)
C
C READ IN COMPONENT COUNTERS
READ(IUIN) (NI(I), I=1,NC)
C
C IF (IOV.GT. 0) GO TO (70,90,110), IOV
C
C ORDERING WILL USE AXIAL COORD.

```

OUTD

OUTD

```

DO 60 I = 1,NF
60 A(I) = AX(I)
   ORD = DORD(1)
   GO TO 130
C
C ORDERING WILL USE X COORD.
76 DO 80 I = 1,NF
80 A(I) = X(I)
   ORD = DORD(2)
   GO TO 130
C
C ORDERING WILL USE Y COORD.
90 DO 100 I = 1,NF
100 A(I) = Y(I)
   ORD = DORD(3)
   GO TO 130
C
C ORDERING WILL USE Z COORD.
110 DO 120 I = 1,NF
120 A(I) = Z(I)
   ORD = DORD(4)
C
C
130 CONTINUE
C FIRST ORDER RANDOM SETS (W,R,T, AXIAL COORD. A)
   N1 = 1
   I = NC
   IF (IOG .LT. 0) GO TO 160
   I = 0
140 I = I + 1
   IF (IOR(I) .EQ. 0) GO TO 230
   N1 = NI(I)
   IF (N1 .EQ. 0) GO TO 230
   IF (I = NC) 150,160,160
150 I2 = I
```

OUTD 071
OUTD 072
OUTD 073
OUTD 074
OUTD 075
OUTD 076
OUTD 077
OUTD 078
OUTD 079
OUTD 080
OUTD 081
OUTD 082
OUTD 083
OUTD 084
OUTD 085
OUTD 086
OUTD 087
OUTD 088
OUTD 089
OUTD 090
OUTD 091
OUTD 092
OUTD 093
OUTD 094
OUTD 095
OUTD 096
OUTD 097
OUTD 098
OUTD 099
OUTD 100
OUTD 101
OUTD 102
OUTD 103
OUTD 104
OUTD 105

OUTD

OUTD

OUTD 106
 OUTD 107
 OUTD 108
 OUTD 109
 OUTD 110
 OUTD 111
 OUTD 112
 OUTD 113
 OUTD 114
 OUTD 115
 OUTD 116
 OUTD 117
 OUTD 118
 OUTD 119
 OUTD 120
 OUTD 121
 OUTD 122
 OUTD 123
 OUTD 124
 OUTD 125
 OUTD 126
 OUTD 127
 OUTD 128
 OUTD 129
 OUTD 130
 OUTD 131
 OUTD 132
 OUTD 133
 OUTD 134
 OUTD 135
 OUTD 136
 OUTD 137
 OUTD 138
 OUTD 139
 OUTD 140

```

155 I2 = I2 + 1
    IF (I2 .GT. NC) GO TO 160
    NT = NI(I2) - 1
    IF (NT) 155, 155, 170
160 NT = NF
170 NT1 = NT - 1
    DO 220 L = NI, NT1
    L1 = L + 1
    J1 = J(L)
C
    DO 210 K = L1, NT
    J2 = J(K)
    IF (A(J1) = A(J2)) 190, 180, 210
C
C TWO EQUAL POINTS
180 GO TO 210
C
190 KD = K - L
    K1 = K
    DO 200 II = 1, KD
    K2 = K1 + 1
    J(K1) = J(K2)
200 K1 = K2
C
    J1 = J2
    J(L) = J2
210 CONTINUE
220 CONTINUE
230 IF (Y .LT. NC) GO TO 140
C
C CHECK IF GROUP ORDERING REQUIRED
    IF (IDG .GT. 0) GO TO 236
    L = 0
    DO 235 I = 1, NT
    I = L + 1
  
```

OUTD

OUTD

OUTD 141
OUTD 142
OUTD 143
OUTD 144
OUTD 145
OUTD 146
OUTD 147
OUTD 148
OUTD 149
OUTD 150
OUTD 151
OUTD 152
OUTD 153
OUTD 154
OUTD 155
OUTD 156
OUTD 157
OUTD 158
OUTD 159
OUTD 160
OUTD 161
OUTD 162
OUTD 163
OUTD 164
OUTD 165
OUTD 166
OUTD 167
OUTD 168
OUTD 169
OUTD 170
OUTD 171
OUTD 172
OUTD 173
OUTD 174
OUTD 175

```
235 JX(L) = J(I)
    GO TO 460
C
C NOW ORDER GROUPS
C FIND BASE COMPONENT
236 DO 240 I = 1,NC
    IB = I
    IF (COMPID(I) .EQ. COMPB) GO TO 250
240 CONTINUE
    IB = 1
C IDENTIFY ELEMENTS OF BASE COMPONENT
250 N1 = NI(IB)
    IF (N1 .GT. 0) GO TO 260
    N1 = 1
    GO TO 290
260 CONTINUE
    IF (IB = NC) 270,290,290
270 IB1 = IB
280 IF (IB1 .GT. NC) GO TO 290
    N2 = NI(IB1) + 1
    IF (N2 .LE. 0) GO TO 280
    GO TO 300
290 N2 = NF
300 NT = N2 + N1 + 1
    L = 0
    DO 310 I = N1,N2
        L = L + 1
310 JX(L) = J(I)
    IF (NC .LE. 1) GO TO 460
    IF (NI(IB) .EQ. 0) GO TO 460
C NOW CYCLE OTHER COMPONENTS
```

OUTD

OUTD

176 OUTD
 177 OUTD
 178 OUTD
 179 OUTD
 180 OUTD
 181 OUTD
 182 OUTD
 183 OUTD
 184 OUTD
 185 OUTD
 186 OUTD
 187 OUTD
 188 OUTD
 189 OUTD
 190 OUTD
 191 OUTD
 192 OUTD
 193 OUTD
 194 OUTD
 195 OUTD
 196 OUTD
 197 OUTD
 198 OUTD
 199 OUTD
 200 OUTD
 201 OUTD
 202 OUTD
 203 OUTD
 204 OUTD
 205 OUTD
 206 OUTD
 207 OUTD
 208 OUTD
 209 OUTD
 210 OUTD

OUTD

```

DO 450 IC = 1, NC
  IF (IC .EQ. JR) GO TO 450
  NC1 = NI(IC)
  IF (NC1 .EQ. 0) GO TO 450
  IF (IC = NC) 320, 340, 340
320 JC1 = IC
330 IC1 = IC1 + 1
  IF (IC1 .GT. NC) GO TO 340
  NC2 = NI(IC1) + 1
  IF (NC2 .LE. 0) GO TO 330
  GO TO 350
340 NC2 = NF
C
350 DO 360 I = NC1, NC2
  L = L + 1
360 JX(L) = J(I)
  IF (JOG .LE. 0) GO TO 450
  LY = L
  J1 = J(NC1)
  DO 370 LX = 1, NT
  L1 = LX
  LP = L1 + 1
  J2 = JX(LX)
  IF (A(J1) = A(J2)) 370, 390, 380
370 CONTINUE
C
C ELEMENTS DO NOT OVERLAP
  NT = LY
  GO TO 450
C
C SAVE REMAINING INDICES FROM BASE COMPONENT
380 L2 = L1
  IF (L2 .GT. NT) GO TO 410
390 LJ1 = JX(L2)
  LJ2 = J(NC2)
  
```

OUTD

OUTD 211
OUTD 212
OUTD 213
OUTD 214
OUTD 215
OUTD 216
OUTD 217
OUTD 218
OUTD 219
OUTD 220
OUTD 221
OUTD 222
OUTD 223
OUTD 224
OUTD 225
OUTD 226
OUTD 227
OUTD 228
OUTD 229
OUTD 230
OUTD 231
OUTD 232
OUTD 233
OUTD 234
OUTD 235
OUTD 236
OUTD 237
OUTD 238
OUTD 239
OUTD 240
OUTD 241
OUTD 242
OUTD 243
OUTD 244
OUTD 245

```
IF (A(LJ1), .EQ., A(LJ2)) NC2 = NC2 - 1
IX = NI - 1
DO 400 LX = L2, NT
  IX = IX + 1
  400 J(IX) = JX(LX)
C
C  INSERT ELEMENTS OF COMPONENT IC
  410 DO 420 I = NC1, NC2
    JX(L1) = J(I)
    420 L1 = L1 + 1
C
C  FILL IN REMAINING BASE ELEMENTS
  IF (L2 .GT. NT) GO TO 440
  DO 430 I = NI, IX
    JX(L1) = J(I)
    430 L1 = L1 + 1
C
C  RESET POINT NUMBERS
  440 L = L1 - 1
    NT = L
C
  450 CONTINUE
  460 CONTINUE
C
  IF (IPRINT .EQ. 1)
    1WRITE(70, 470) ORD, IOG, (IOR(I), I=1, 10)
    470 FORMAT(1H0, 19HPPOINTS ORDERED FOR , A1, 14H AS PER IOG = , I3,
      1 14H AND IOR(I) = , 10I3/1H0, T14, 2H10, 7X1HJ, 742, 1HX,
      2 762, 1HY, 782, 1HZ, T102, 1HR, T122, 1HA/)
C
  DO 490 L = 1, NT
    I = JX(L)
    JJE = RJ(I)
    IC = ID(I)
    IF (IPRINT .EQ. 1)
```

OUTD

OUTD

```
      1WRITE(10,480)COMPID(IC),JJ,X(I),Y(I),Z(I),R(I),AX(I)
      480 FOPMAT(1H , 1I2,A4, 4X14, 4X.5F20.6)
      490 CONTINUE
C
C
      500 CONTINUE
C
      RETURN
      END
```

```
OUTD 246
OUTD 247
OUTD 248
OUTD 249
OUTD 250
OUTD 251
OUTD 252
OUTD 253
OUTD 254
OUTD 255
```