MODULE MESHDATA

!

! MODULE STORING DATA DEFINING THE MESH.

!

REAL :: X(10000),Y(10000),AI(20000),AJ(20000),AK(20000)

REAL :: BI(20000),BJ(20000),BK(20000),AREA(20000)

INTEGER :: NEL,NNP,NBP,MAXNEL,MAXNNP,MAXNBP

INTEGER :: NPI(20000),NPJ(20000),NPK(20000),NPB(400)

INTEGER :: NPTS1,NPTS2,NPTS3,NPTS4

END MODULE MESHDATA

MODULE EQNSDATA

!

! MODULE STORING EQUATION DATA: RECTANGULARISED OVERALL STIFFNESS

! MATRIX, FORCE VECTOR AND SOLUTION VECTOR.

!

REAL :: OSTIFF(10000,13),F(10000),DELTA(10000)

INTEGER :: NPA(10000,13),NAP(10000)

END MODULE EQNSDATA

PROGRAM HARMON

!

! PROGRAM FOR FINITE ELEMENT ANALYSIS OF TWO-DIMENSIONAL PROBLEMS OF

! THE HARMONIC TYPE, USING CONSTANT STRAIN (RATE) TRIANGULAR

! ELEMENTS.

!

USE MESHDATA

USE EQNSDATA

REAL :: PHI1(20000),B(2,3),ESTIFF(3,3)

INTEGER :: IJK(3)

CHARACTER(80) :: TITLE

OPEN(5,FILE="DATA")

OPEN(6,FILE="RESULTS")

OPEN(7,FILE="MESHRES")

!

! DEFINE MAXIMUM PROBLEM SIZE PERMITTED BY THE ARRAY DIMENSIONS.

MAXNEL=20000

MAXNNP=10000

MAXNBP=400

!

! INPUT THE PROBLEM TITLE.

READ(5,FMT="(A80)") TITLE

WRITE(6,61) TITLE

61 FORMAT("CST FINITE ELEMENT SOLUTION FOR TWO-DIMENSIONAL",

& " HARMONIC PROBLEM" // A)

!

! INPUT OR GENERATE THE MESH DATA AND PHI FUNCTION (FOR THE HARMONIC

! EQUATION).

CALL MESH

CALL MODIFY

CALL PHI1F(PHI1)

!

! COMPUTE THE ELEMENT GEOMETRIES.

Each element in turn: DO M=1,NEL

I=NPI(M)

J=NPJ(M)

K=NPK(M)

AI(M)=-X(J)+X(K)

AJ(M)=-X(K)+X(I)

AK(M)=-X(I)+X(J)

BI(M)=Y(J)-Y(K)

BJ(M)=Y(K)-Y(I)

BK(M)=Y(I)-Y(J)

AREA(M)=0.5\*(AK(M)\*BJ(M)-AJ(M)\*BK(M))

IF(AREA(M) <= 0.) THEN

WRITE(6,62) M

62 FORMAT(/"ELEMENT NUMBER",I6," HAS NEGATIVE AREA - STOP")

STOP

END IF

END DO Each element in turn

!

! OUTPUT THE MESH GEOMETRY DATA.

CALL MSHOUT

!

! SET INITIAL VALUES OF STIFFNESSES, EXTERNAL FORCES AND UNKNOWNS.

Each overall row in turn: DO IROW=1,NNP

Each overall column in turn: DO IC=1,13

OSTIFF(IROW,IC)=0.

NPA(IROW,IC)=0

END DO Each overall column in turn

NPA(IROW,1)=IROW

F(IROW)=0.

DELTA(IROW)=0.

END DO Each overall row in turn

!

! SET UP THE OVERALL ASSEMBLY LOOP.

Each element in turn: DO M=1,NEL

!

! STORE THE ELEMENT NODE NUMBERS IN ORDER IN ARRAY IJK.

IJK(1)=NPI(M)

IJK(2)=NPJ(M)

IJK(3)=NPK(M)

!

! COMPUTE THE EXTERNAL FORCE COMPONENTS ON EACH NODE OF THE ELEMENT.

FM=-PHI1(M)\*AREA(M)/3.

!

! FORM THE ELEMENT STIFFNESS MATRIX.

B(1,1)=BI(M)

B(1,2)=BJ(M)

B(1,3)=BK(M)

B(2,1)=AI(M)

B(2,2)=AJ(M)

B(2,3)=AK(M)

FACT=0.25/AREA(M)

Each element row in turn: DO IRE=1,3

Each element column in turn: DO ICE=1,3

ESTIFF(IRE,ICE)=FACT\*(B(1,IRE)\*B(1,ICE)+B(2,IRE)\*B(2,ICE))

!

! ADD ELEMENT STIFFNESS TO OVERALL STIFFNESS.

IROW=IJK(IRE)

ICOL=IJK(ICE)

!

! STORE STIFFNESS COEFFICIENT IN RECTANGULAR FORM OF OVERALL MATRIX.

IFLAG=0

Each overall column in turn: DO IC=1,13

IF(NPA(IROW,IC) == 0) THEN

NPA(IROW,IC)=ICOL

NAP(IROW)=IC

END IF

IF(NPA(IROW,IC) == ICOL) THEN

OSTIFF(IROW,IC)=OSTIFF(IROW,IC)+ESTIFF(IRE,ICE)

IFLAG=1

EXIT

END IF

END DO Each overall column in turn

IF(IFLAG == 0) THEN

WRITE(6,63) IROW

63 FORMAT(/"NODE ",I6," HAS MORE THAN 12 ADJACENT NODES - STOP")

STOP

END IF

!

END DO Each element column in turn

END DO Each element row in turn

!

! ASSEMBLE THE EXTERNAL FORCES ON THE NODES.

Each element row in turn: DO IRE=1,3

IROW=IJK(IRE)

F(IROW)=F(IROW)+FM

END DO Each element row in turn

!

END DO Each element in turn

!

! APPLY THE BOUNDARY CONDITIONS.

CALL BCS

!

! SOLVE THE LINEAR EQUATIONS.

CALL SOLVE1

!

! OUTPUT THE REQUIRED RESULTS.

CALL OUTPUT

STOP

END PROGRAM HARMON

SUBROUTINE MESH

!

! SUBPROGRAM TO READ OR GENERATE A MESH OF TRIANGULAR FINITE ELEMENTS.

! THIS VERSION READS IN THE NECESSARY DATA.

!

USE MESHDATA

!

! INPUT THE NUMBERS OF NODES AND ELEMENTS.

READ(5,\*) NNP,NEL

IF(NNP > MAXNNP .OR. NEL > MAXNEL) THEN

WRITE(6,61) NNP,NEL

61 FORMAT(/"EXCESSIVE SIZE OF MESH, NNP =",I6,", NEL =",I6)

STOP

END IF

!

! INPUT THE NODAL POINT CO-ORDINATES.

READ(5,\*) (I,X(I),Y(I),N=1,NNP)

!

! INPUT THE ELEMENT NODE DATA.

READ(5,\*) (M,NPI(M),NPJ(M),NPK(M),N=1,NEL)

RETURN

END SUBROUTINE MESH

SUBROUTINE MESH

!

! SUBPROGRAM TO READ OR GENERATE A MESH OF TRIANGULAR FINITE ELEMENTS.

! THIS VERSION GENERATES A SQUARE MESH OF RIGHT-ANGLED TRIANGLES.

!

USE MESHDATA, NXPT=>NPTS1, NYPT=>NPTS2

!

! INPUT AND STORE THE NUMBERS OF POINTS REQUIRED IN THE X AND Y

! DIRECTIONS.

READ(5,\*) NXPT,NYPT

!

! COMPUTE AND TEST THE NUMBERS OF NODES AND ELEMENTS.

NNP=NXPT\*NYPT

NEL=(NXPT-1)\*(NYPT-1)\*2

IF(NNP > MAXNNP .OR. NEL > MAXNEL) THEN

WRITE(6,61) NNP,NEL

61 FORMAT(/"EXCESSIVE SIZE OF MESH, NNP =",I6,", NEL =",I6)

STOP

END IF

!

! DEFINE THE NODAL POINT CO-ORDINATES.

Each horizontal row of nodes in turn: DO IY=1,NYPT

Each node along the row in turn: DO IX=1,NXPT

I=(IY-1)\*NXPT+IX

X(I)=FLOAT(IX-1)/FLOAT(NXPT-1)

Y(I)=FLOAT(IY-1)/FLOAT(NYPT-1)

END DO Each node along the row in turn

END DO Each horizontal row of nodes in turn

!

! DEFINE THE NUMBERS OF THE THREE NODES OF EACH ELEMENT.

NXEL=NXPT-1

NYEL=NYPT-1

Each horizontal row of elements in turn: DO IY=1,NYEL

Each pair of elements along the row in turn: DO IX=1,NXEL

NSQ=(IY-1)\*NXEL+IX

M1=NSQ\*2-1

M2=M1+1

I=(IY-1)\*NXPT+IX

NPI(M1)=I

NPJ(M1)=I+NXPT+1

NPK(M1)=I+NXPT

NPI(M2)=I

NPJ(M2)=I+1

NPK(M2)=I+1+NXPT

END DO Each pair of elements along the row in turn

END DO Each horizontal row of elements in turn

RETURN

END SUBROUTINE MESH

SUBROUTINE MESH

!

! SUBPROGRAM TO READ OR GENERATE A MESH OF TRIANGULAR FINITE ELEMENTS.

! THIS VERSION GENERATES A UNIFORM EQUILATERAL TRIANGULAR MESH.

!

USE MESHDATA, NSPT=>NPTS1

!

! INPUT AND STORE THE NUMBER OF POINTS ON EACH SIDE OF THE MESH.

READ(5,\*) NSPT

!

! COMPUTE AND TEST THE NUMBERS OF NODES AND ELEMENTS.

NNP=NSPT\*(NSPT+1)/2

NEL=(NSPT-1)\*\*2

IF(NNP > MAXNNP .OR. NEL > MAXNEL) THEN

WRITE(6,61) NNP,NEL

61 FORMAT(/"EXCESSIVE SIZE OF MESH, NNP =",I6,", NEL =",I6)

STOP

END IF

!

! DEFINE THE NODAL POINT CO-ORDINATES.

HX=1./FLOAT(NSPT-1)

HY=HX\*0.5\*SQRT(3.)

I=0

Each horizontal row of nodes in turn: DO IY=1,NSPT

NXPT=NSPT-IY+1

Each node along the row in turn: DO IX=1,NXPT

I=I+1

X(I)=FLOAT(IX-1)\*HX+FLOAT(IY-1)\*0.5\*HX

Y(I)=FLOAT(IY-1)\*HY

END DO Each node along the row in turn

END DO Each horizontal row of nodes in turn

!

! DEFINE THE NUMBERS OF THE THREE NODES OF EACH ELEMENT,

! FIRST FOR THE UPWARD POINTING ELEMENTS.

M=0

NYEL=NSPT-1

Each horizontal row of elements in turn: DO IY=1,NYEL

NXEL=NSPT-IY

Each element along the row in turn: DO IX=1,NXEL

M=M+1

NPI(M)=M+IY-1

NPJ(M)=NPI(M)+1

NPK(M)=M+NSPT

END DO Each element along the row in turn

END DO Each horizontal row of elements in turn

!

! THEN FOR THE DOWNWARD POINTING ELEMENTS.

M1=M

NYEL=NYEL-1

Each horizontal row of elements in turn: DO IY=1,NYEL

NXEL=NSPT-IY-1

Each element along the row in turn: DO IX=1,NXEL

M=M+1

NPI(M)=M-M1+2\*IY-1

NPJ(M)=M-M1+NSPT+IY

NPK(M)=NPJ(M)-1

END DO Each element along the row in turn

END DO Each horizontal row of elements in turn

RETURN

END SUBROUTINE MESH

SUBROUTINE MESH

!

! SUBPROGRAM TO READ OR GENERATE A MESH OF TRIANGULAR FINITE ELEMENTS.

! THIS VERSION GENERATES A SQUARE MESH OF MAINLY ISOSCELES ELEMENTS.

!

USE MESHDATA, NXPT=>NPTS1, NYPT=>NPTS2

!

! INPUT AND STORE THE NUMBERS OF POINTS ALONG THE X AND Y AXES.

READ(5,\*) NXPT,NYPT

!

! COMPUTE AND TEST THE NUMBERS OF NODES AND ELEMENTS.

MODNY=MOD(NYPT,2)

IF(MODNY == 0) NNP=NYPT\*(2\*NXPT+1)/2

IF(MODNY == 1) NNP=(NYPT-1)\*(2\*NXPT+1)/2+NXPT

NEL=(NYPT-1)\*(2\*NXPT-1)

IF(NNP > MAXNNP .OR. NEL > MAXNEL) THEN

WRITE(6,61) NNP,NEL

61 FORMAT(/"EXCESSIVE SIZE OF MESH, NNP =",I6,", NEL =",I6)

STOP

END IF

!

! DEFINE THE NODAL POINT CO-ORDINATES.

I=0

Each horizontal row of nodes in turn: DO IY=1,NYPT

MODIY=MOD(IY,2)

Each node along the row in turn: DO IX=1,NXPT

I=I+1

X(I)=FLOAT(IX-1)/FLOAT(NXPT-1)

Y(I)=FLOAT(IY-1)/FLOAT(NYPT-1)

IF(MODIY == 0 .AND. IX > 1) X(I)=X(I)-0.5/FLOAT(NXPT-1)

END DO Each node along the row in turn

IF(MODIY == 0) THEN

I=I+1

Y(I)=Y(I-1)

X(I)=1.

END IF

END DO Each horizontal row of nodes in turn

!

! DEFINE THE NUMBERS OF THE THREE NODES OF EACH ELEMENT,

! FIRST FOR THE UPWARD POINTING ELEMENTS.

M=0

NYEL=NYPT-1

Each horizontal row of elements in turn: DO IY=1,NYEL

NXEL=NXPT-1

IF(MOD(IY,2) == 0) NXEL=NXPT

Each element along the row in turn: DO IX=1,NXEL

M=M+1

NPI(M)=M+IY-1

NPJ(M)=NPI(M)+1

NPK(M)=NPJ(M)+NXPT

END DO Each element along the row in turn

END DO Each horizontal row of elements in turn

!

! THEN FOR THE DOWNWARD POINTING ELEMENTS.

M1=M

Each horizontal row of elements in turn: DO IY=1,NYEL

NXEL=NXPT

IF(MOD(IY,2) == 0) NXEL=NXPT-1

Each element along the row in turn: DO IX=1,NXEL

M=M+1

NPI(M)=M-M1+IY-1

NPJ(M)=NPI(M)+NXPT+1

NPK(M)=NPJ(M)-1

END DO Each element along the row in turn

END DO Each horizontal row of elements in turn

RETURN

END SUBROUTINE MESH

SUBROUTINE MESH

!

! SUBPROGRAM TO READ OR GENERATE A MESH OF TRIANGULAR FINITE ELEMENTS.

! THIS VERSION GENERATES A CIRCULAR MESH.

!

USE MESHDATA, NCEL=>NPTS1, NRPT=>NPTS2

!

! INPUT NUMBER OF ELEMENTS AT CENTRE AND POINTS ALONG A RADIUS.

READ(5,\*) NCEL,NRPT

!

! COMPUTE AND TEST THE NUMBERS OF NODES AND ELEMENTS.

NNP=NCEL\*NRPT\*(NRPT-1)/2+1

NEL=NCEL\*(NRPT-1)\*\*2

IF(NNP > MAXNNP .OR. NEL > MAXNEL) THEN

WRITE(6,61) NNP,NEL

61 FORMAT(/"EXCESSIVE SIZE OF MESH, NNP =",I6,", NEL =",I6)

STOP

END IF

!

! DEFINE THE NODAL POINT CO-ORDINATES.

X(1)=0.

Y(1)=0.

PI=4.\*ATAN(1.)

I=1

NREL=NRPT-1

Each ring of nodes in turn: DO IR=1,NREL

R=FLOAT(IR)/FLOAT(NREL)

NTHPT=NCEL\*IR

Each node along the ring in turn: DO ITH=1,NTHPT

THETA=FLOAT(ITH-1)\*2.\*PI/FLOAT(NTHPT)

I=I+1

X(I)=R\*COS(THETA)

Y(I)=R\*SIN(THETA)

END DO Each node along the ring in turn

END DO Each ring of nodes in turn

!

! DEFINE THE NUMBERS OF THE THREE NODES OF EACH ELEMENT.

!

! INWARD POINTING ELEMENTS.

M=0

I=1

Each ring of inward pointing elements in turn: DO IR=1,NREL

NTHPT=NCEL\*IR

Each element along the ring in turn: DO ITH=1,NTHPT

M=M+1

IF(ITH == 1) NPI(M)=I

IF(ITH > 1) NPI(M)=NPI(M-1)+1

IF(ITH > 1 .AND. MOD(ITH-1,IR) == 0) NPI(M)=NPI(M-1)

NPJ(M)=M+1

NPK(M)=M+2

IF(ITH == 1) THEN

I=NPI(M)

K=NPJ(M)

END IF

END DO Each element along the ring in turn

NPI(M)=I

NPK(M)=K

I=K

END DO Each ring of inward pointing elements in turn

!

! OUTWARD POINTING ELEMENTS.

M1=M

J=NCEL+3

Each ring of outward pointing elements in turn: DO IR=2,NREL

NTHPT=NCEL\*(IR-1)

Each element along the ring in turn: DO ITH=1,NTHPT

M=M+1

NPI(M)=M-M1+1

IF(ITH == 1) NPJ(M)=J

IF(ITH > 1) NPJ(M)=NPJ(M-1)+1

IF(ITH > 1 .AND. MOD(ITH-1,IR-1) == 0) NPJ(M)=NPJ(M-1)+2

NPK(M)=NPI(M)+1

IF(ITH == 1) K=NPI(M)

END DO Each element along the ring in turn

NPK(M)=K

J=NPJ(M)+2

END DO Each ring of outward pointing elements in turn

RETURN

END SUBROUTINE MESH

SUBROUTINE MODIFY

!

! SUBPROGRAM TO MODIFY THE MESH.

! THIS VERSION APPLIES LINEAR SCALING TO THE NODE CO-ORDINATES.

!

USE MESHDATA

!

! INPUT THE DEPTH (Y-DIRECTION) AND WIDTH (X-DIRECTION).

READ(5,\*) H,W

!

! MODIFY THE CO-ORDINATES OF THE NODAL POINTS.

Each node in turn: DO I=1,NNP

X(I)=X(I)\*W

Y(I)=Y(I)\*H

END DO Each node in turn

RETURN

END SUBROUTINE MODIFY

SUBROUTINE PHI1F(PHI1)

!

! SUBPROGRAM TO DEFINE THE MEAN VALUE OF THE PHI1 FUNCTION IN THE

! HARMONIC DIFFERENTIAL EQUATION FOR EACH ELEMENT IN THE MESH.

!

USE MESHDATA

!

REAL :: PHI1(20000)

READ(5,\*) PZ,VISCOS

WRITE(6,61) PZ,VISCOS

61 FORMAT(/"PRESSURE GRADIENT =",E12.4,10X,"VISCOSITY =",E12.4)

RATIO=PZ/VISCOS

Each element in turn: DO M=1,NEL

PHI1(M)=RATIO

END DO Each element in turn

RETURN

END SUBROUTINE PHI1F

SUBROUTINE MSHOUT

!

! SUBPROGRAM TO WRITE OUT THE GEOMETRIC DATA FOR THE MESH.

!

USE MESHDATA

!

! OUTPUT THE NUMBER OF ELEMENTS, NODAL POINTS AND CO-ORDINATES.

WRITE(7,71) NEL,NNP,(I,X(I),Y(I),I=1,NNP)

71 FORMAT(/"GEOMETRIC DATA FOR THE MESH" //

& 10X,"NUMBER OF ELEMENTS = ",I6 //

& 10X,"NUMBER OF NODAL POINTS = ",I6 //

& "NODAL POINT CO-ORDINATES" //

& 3(" I X Y ")/(3(I7,2F9.4)))

!

! OUTPUT THE ELEMENT NODE NUMBERS AND AREAS.

WRITE(7,72) (M,NPI(M),NPJ(M),NPK(M),AREA(M),M=1,NEL)

72 FORMAT(/"ELEMENT NODE NUMBERS AND AREAS" //

& 2(" M I J K AREA ")/

& (2(4I7,E12.4)))

RETURN

END SUBROUTINE MSHOUT

SUBROUTINE BCS

!

! SUBPROGRAM TO APPLY THE BOUNDARY CONDITIONS.

! THIS VERSION PRESCRIBES ZERO VALUES OF THE UNKNOWNS.

!

USE MESHDATA

USE EQNSDATA

!

! INPUT THE BOUNDARY NODE NUMBERS.

READ(5,\*) NBP

IF(NBP > MAXNBP) THEN

WRITE(6,61) NBP

61 FORMAT("EXCESSIVE NUMBER OF BOUNDARY POINTS, NBP = ",I5)

STOP

END IF

READ(5,\*) (NPB(I),I=1,NBP)

!

! APPLY ZERO VALUES OF THE UNKNOWNS AT THE BOUNDARY POINTS.

FACT=1.E10

Each boundary point in turn: DO I=1,NBP

IROW=NPB(I)

OSTIFF(IROW,1)=OSTIFF(IROW,1)\*FACT

F(IROW)=0.

END DO Each boundary point in turn

!

! OUTPUT THE BOUNDARY POINT NUMBERS.

WRITE(7,71) NBP,(NPB(IB),IB=1,NBP)

71 FORMAT(/"THE NUMBERS OF THE ",I3," BOUNDARY POINTS ARE" /

& (10I8))

RETURN

END SUBROUTINE BCS

SUBROUTINE SOLVE1

!

! SUBPROGRAM FOR SOLVING BY GAUSS-SEIDEL METHOD THE LINEAR EQUATIONS

! OBTAINED FROM THE FINITE ELEMENT FORMULATION OF HARMONIC PROBLEMS.

!

USE EQNSDATA

USE MESHDATA

!

! INPUT THE SOLUTION PARAMETERS.

READ(5,\*) NCYCLE,IFREQ,ORELAX,TOLER

WRITE(6,61) ORELAX

61 FORMAT(/"SOLUTION OF EQUATIONS BY GAUSS-SEIDEL ITERATION" //

& "OVER-RELAXATION FACTOR = ",F6.3)

!

! SET UP ITERATION LOOP.

IF(IFREQ /= 0) WRITE(\*,62)

62 FORMAT(" ITER ERROR ")

Each cycle of iteration in turn: DO ITER=1,NCYCLE

SUMD=0.

SUMDD=0.

!

! OBTAIN NEW ESTIMATE FOR EACH UNKNOWN IN TURN.

Each equation in turn: DO IROW=1,NNP

DELD=F(IROW)

ICMAX=NAP(IROW)

Each overall column in turn: DO IC=1,ICMAX

ICOL=NPA(IROW,IC)

DELD=DELD-OSTIFF(IROW,IC)\*DELTA(ICOL)

END DO Each overall column in turn

DELD=DELD/OSTIFF(IROW,1)

SUMDD=SUMDD+ABS(DELD)

DELTA(IROW)=DELTA(IROW)+DELD\*ORELAX

SUMD=SUMD+ABS(DELTA(IROW))

END DO Each equation in turn

!

! TEST FOR CONVERGENCE.

ERROR=SUMDD/SUMD

IF(ERROR < TOLER) EXIT

!

! OUTPUT PROGRESS INFORMATION EVERY IFREQ CYCLES, UNLESS IFREQ=0.

IF(IFREQ /= 0) THEN

IF(MOD(ITER,IFREQ) == 0) WRITE(\*,63) ITER,ERROR

63 FORMAT(I8,E15.4)

END IF

!

END DO Each cycle of iteration in turn

!

! WARN OF FAILURE TO CONVERGE.

IF(ERROR > TOLER) THEN

WRITE(6,64) NCYCLE

64 FORMAT(/"NO CONVERGENCE AFTER",I8," CYCLES")

RETURN

END IF

!

! OUTPUT NUMBER OF ITERATIONS AND TOLERANCE FOR CONVERGED SOLUTION.

WRITE(6,65) TOLER,ITER

65 FORMAT(/"ITERATION CONVERGED TO A TOLERANCE OF",E12.4,

& " AFTER",I8," CYCLES")

RETURN

END SUBROUTINE SOLVE1

SUBROUTINE OUTPUT

!

! SUBPROGRAM TO OUTPUT THE FINAL RESULTS.

!

USE MESHDATA

USE EQNSDATA

!

! INTEGRATE OVER THE SOLUTION DOMAIN.

SUM=0.

Each element in turn: DO M=1,NEL

I=NPI(M)

J=NPJ(M)

K=NPK(M)

DMEAN=(DELTA(I)+DELTA(J)+DELTA(K))/3.

SUM=SUM+DMEAN\*AREA(M)

END DO Each element in turn

WRITE(6,61) SUM

61 FORMAT(/"INTEGRAL OVER THE SOLUTION DOMAIN =",E14.6)

RETURN

END SUBROUTINE OUTPUT