รายการอ้างอิง

1. Chapra, S. C. and Canale, R. P. Numerical Methods for Engineers. Second Edition. New York : McGraw-Hill, 1989.

2. ปราโมทย์ เดชะอำไพ. ไฟไนต์เอลิเมนต์ในงานวิศวกรรม. พิมพ์ครั้งที่ 2. กรุงเทพฯ : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2542.

3. Anderson, J. D. Jr. Computational Fluid Dynamics: The Basic with Application. McGraw-Hill Series in Mechanical Engineering. Singapore : McGraw-Hill, 1995.

4. Huebner, H. K., Thornton, E. A. and Byrom, T. G. The Finite Element Method for Engineers. Third Edition. New York : John Wiley & Sons, Inc., 1995.

5. Zienkiewicz, O. C. and Taylor, R. L. The Finite Element Method. Fourth Edition. New York : McGraw-Hill, 1991.

6. White, F. M. Viscous Fluid Flow. Second Edition. McGraw-Hill Series in Mechanical Engineering. Singapore : McGraw-Hill, 1991.

7. Reddy, J. N. and Satake, A. A comparison of a penalty finite element model with the stream function-vorticity model of natural convection in enclosures. Journal of Heat Transfer 102 (1980) : 659-666.

8. Runchal, A. K. Convergence and accuracy of three finite difference schemes for a two-dimensional conduction and convection problem. International Journal for Numerical Methods in Engineering 4 (1972) : 541-550.

9. Patankar, S. V. Numerical Heat Transfer and Fluid Flow. Series in Computational Methods in Mechanics and Thermal Sciences. New York : Hemisphere, 1980.

10. ปราโมทย์ เดชะอำไพ. ระเบียบวิธีเชิงตัวเลขในงานวิศวกรรม. พิมพ์ครั้งที่ 2. กรุงเทพฯ : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2541.

11. Chorin, A. J. Numerical solution of the Navier-Stokes equations. Mathematics of Computation. 22 (1968) : 745-762.

12. Patankar, S. V. and Spalding, D. B. A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-dimensional Parabolic Flows. International Journal of Heat and Mass Transfer 15 (1972) : 1787.

13. Versteeg, H. K. and Malalasekera, W. An Introduction to Computational Fluid Dynamics: The Finite Volume Method. London : Longman Scientific & Technical, 1995.

14. Reddy, J. N. and Gartling, D. K. The Finite Element Method in Heat Transfer and Fluid Dynamics. Florida : CRC Press, 1994.

15. จิตดิน ตรีพุทธรัตน์. การศึกษาการไหลผ่านวัตถุด้วยวิธีการไฟไนต์เอลิเมนต์. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2539.

16. สุพัฒนพงศ์ สิขาบัณฑิต. เทคนิคการปรับขนาดไฟไนต์เอลิเมนต์เอลิเมนต์เพื่อการวิเคราะห์การไหลแบบหนืด. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2541.

17. วรสิทธิ์ กาญจนกิจเกษม. ระเบียบวิธีไฟไนต์เอลิเมนต์สำหรับการไหลแบบไม่อัดตัวชนิดหนืดที่สภาวะอยู่ตัว. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2541.

18. Comini, G. and Giudice, S. D. Finite-element solution of the incompressible Navier-Stokes equations. Numerical Heat Transfer 5 (1982) : 463-478.

19. Shaw, C. T. Using a segregated finite element scheme to solve the incompressible Navier-Stokes equations. International Journal for Numerical Methods in Fluids 12 (1991) : 81-92.

20. Baliga, B. R. and Patankar, S. V. A control volume finite-element method for two-dimensional fluid flow and heat transfer. Numerical Heat Transfer 6 (1983) : 245-261.

21. Baliga, B. R. and Patankar, S. V. A new finite-element formualtion for convection-diffusion problems. Numerical Heat Transfer 3 (1980) : 393-409.

22. Prakash, C. and Patankar, S. V. A control volume-based finite-element method for solving the Navier-Stokes equations using equal-order velocity-pressure interpolation. Numerical Heat Transfer 8 (1985) : 259-280.

23. Schnipke, R. J. and Rice, J. G. Examination of a new finite element method applied to convection heat transfer. Finite Elements in Analysis and Design 1 (1985) : 227-239.

24. Rice, J. G. and Schnipke, R. J. An equal-order velocity-pressure formulation that does not exhibit spurious pressure modes. Computer Methods in Applied Mechanics and Engineering 58 (1986) : 135-149.

25. Schnipke, R. J. and Rice, J. G. A finite element method for free and forced convection heat transfer. International Journal for Numerical Methods in Engineering 24 (1987) : 117-128.

26. Spalding, D. B. A noval finite-difference formulation for differential expressions involving both first and second derivatives. International journal for Numerical Methods in Engineering 4 (1972) : 551.

27. Leonard, B. P. A stable and accurate convective modeling procedure based on quadratic upstream interpolation. Computer Methods in Applied Mechanics and Engineering 19 (1979) : 59-98.

28. Brooks, A. N. and Hughes, T. J. R. Streamline Upwind/Petrov-Galerkin formulation for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. Computer Methods in Applied Mechanics and Engineering 32 (1982) : 199-259.

29. Rice, J. G. and Schnipke, R. J. A monotone streamline upwind finite element method for convection-dominated flow. Computer Methods in Applied Mechanics and Engineering 48 (1985) : 313-327.

30. Schnipke, Rita Jane. A Streamline Upwind Finite Element Method For Laminar and Turbulent Flow. Ph.D. Dissertation. Mechanical Engineering. The School Of Engineering And Applied Science. University Of Virginia, 1986.

31. Dechaumphai, P. Adaptive finite element technique for heat transfer problems. Energy, Heat & Mass Transfer 17, (1995) : 87-94.

32. Dechaumphai, P and Janphaisaeng, P. Adaptive finite element technique for high-speed compressible flows. Thammasat International Journal of Science and Technology Vol. 3. No. 1, 1998.

33. Etter, D. M. Fortran 77 with Numerical Methods for Engineers and Scientists. California : The Benjamin/Cummings, 1992.

34. Hill, D. L. and Baskharone, E. A. A monotone streamline upwind method for Quadratic finite elements. International Journal for Numerical Methods in Fluid 17 (1993) : 463-475.

35. Raithby, G. D. Skew upstream differencing schemes for problems involving fluid flow. Computer Methods in Applied Mechanics and Engineering 9 (1976) : 153-164.

36. Hassan, Y. A., Rice, J. G. and Kim, J. H. A stable mass-flow-weighted two-dimensional skew upwind scheme. Numerical Heat Transfer 6 (1983) : 395-408.

37. Smith, R. M. and Hutton, A. G. The numerical treatment of advection: A performance comparison of current methods. Numerical Heat Transfer 5, (1982) : 439-461.

38. Fletcher, C. A. J. Computational Techniques for Fluid Dynamics 1. 2-Volume Set. Spring Series in Computational Physics. New York: Springer-Verlag, 1988.

39. Brown, G. M. AIChE Journal 6 (1960) : 179-183.

40. Shewchuck, J. R. An introduction to the conjugate gradient method without the agonizing pain. Pittsburgh, PA 15213, Carnegie Mellon University, 1994. Available from WARP.CS.CMU.EDU ; INTERNET.

41. Hestense, M. R. and Stiefel, E. Methods of conjugate gradients for solving linear system. Journal of Research of the National Bureau of Standards Vol. 49 No. 6 (December, 1952) : 409-436.

42. Golub, G. H. and VanLoan, C. F. Matrix Computations. Baltimore, Maryland : John Hopkins University Press, 1983.

43. Kelly, C. T. Iterative methods for linear and nonlinear equation. Society for Industrial and Applied Mathematics, Vol. 16, Philadelphia, PA., 1995.

44. Fox, R. W. and McDonald, A. T. Introduction to Fluid Mechanics. Fourth Edition. New York : John Wiley & Sons, Inc., 1994.

45. Schlichting, H. Boundary-Layer Theory. Seventh Edition. New York : McGraw-Hill, 1978.

46. ดร.วริทธิ์ อึ้งภากรณ์. การออกแบบเครื่องจักรกล. เล่ม 2. กรุงเทพฯ : ซีเอ็ดยูเคชั่น, 2521.

47. Shah, R. K. and London, A. L. Laminar Flow Forced Convection in Ducts. New York : Acadamic, 1978.

48. Ramaswamy, B. and Jue, T. C. A Segregated finite element formulation of Navier-Stokes equations under laminar conditions. Finite Elements in Analysis and Design 9 (1991) : 257-270.

49. Armaly, B. F., Durst, F., Pereira, J. G. F. and Schonung, B. Experimental and theoretical investigation of backward-facing step flow. Journal of Fluid Mechanics 127 (1983) : 473-496.

50. Taneda, S. Experimental investigation of the wakes behind cylinders and plates at low Reynolds numbers. J. Phys. Soc. Japan 11 (1956) : 302-307.

51. Shames, I. H. Introduction to Solid Mechanics. Second Edition. New Jersey : Prentice-Hall, 1989.

ภาคผนวก

ภาคผนวก ก

รายละเอียดของโปรแกรม EQUAL

โปรแกรมคอมพิวเตอร์ EQUAL ที่ประดิษฐ์ขึ้นดังที่ได้กล่าวไว้ในบทที่ 6 มีรายละเอียดดังนี้

```
C-------------------------------------------------------------
C
C     A FINITE ELEMENT PROGRAM FOR VISCOUS FLOW ANALYSIS
C     USING EQUAL-ORDER FORMULATION
C
C-------------------------------------------------------------

      PARAMETER (MXPOI=3500, MXELE=6400)
      PARAMETER (MXNEQ=MXPOI)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION COORD(MXPOI,2), TEXT(20)
      DIMENSION U(MXPOI),    V(MXPOI),    P(MXPOI),   ASYS(MXNEQ,MXNEQ)
      DIMENSION UOLD(MXPOI), VOLD(MXPOI), RU(MXPOI),  RV(MXPOI)
      DIMENSION UHAT(MXPOI), VHAT(MXPOI), CKP(MXPOI), POLD(MXPOI)
      DIMENSION AREAVG(MXPOI), VSUM(MXPOI)

      CHARACTER*20 NAME1, NAME2, NAME3

      INTEGER  INTMAT(MXELE,3), INTBOU(MXELE,2)
      INTEGER  IBCU(MXPOI), IBCV(MXPOI), IBCP(MXPOI), IBCE(MXPOI)

      INTEGER(2) sthour, stminute, stsecond, sthund
      INTEGER(2) enhour, enminute, ensecond, enhund

C-------------------------------------------------------------

   10 WRITE(6,20)
   20 FORMAT(/, ' PLEASE ENTER INPUT FILE NAME:',/)
      READ(5, '(A)', ERR=10) NAME1

      OPEN(UNIT=7, FILE=NAME1, STATUS='OLD', ERR=10)

C-------------------------------------------------------------
      CALL GETTIM(sthour, stminute, stsecond, sthund)
C-------------------------------------------------------------

C---------------------------------
C     READ TEXT
C---------------------------------
      READ(7,*) NLINES
      DO 100 ILINE=1,NLINES
      READ(7,1) TEXT
    1 FORMAT(20A4)
  100 CONTINUE

C---------------------------------
C     READ INPUT DATA
C---------------------------------
      READ(7,1) TEXT
      READ(7,*) NPOI, NELEM, NBOU, NITER, TOL

      IF(NPOI.GT.MXPOI)  WRITE(6,110) NPOI
  110 FORMAT(/,' PLEASE INCREASE THE PARAMETER MXPOI TO',I5)
      IF(NPOI.GT.MXPOI)  STOP

      IF(NELEM.GT.MXELE) WRITE(6,120) NELEM
  120 FORMAT(/,' PLEASE INCREASE THE PARAMETER MXELE TO',I5)
      IF(NELEM.GT.MXELE) STOP

C---------------------------------
C     READ PROPERTIES OF FLUID
C---------------------------------
      READ(7,1) TEXT
      READ(7,*) DEN, VIS
```

```
C-------------------------------------------------------------
C     READ NODAL COORDINATES, BOUNDARY CONDITIONS, THEIR VALUES
C-------------------------------------------------------------
      READ(7,1) TEXT

      DO 150 IP = 1,NPOI

      READ(7,*) I, IBCU(I), IBCV(I), IBCP(I),
     *          (COORD(I,K), K=1,2), U(I), V(I), P(I)

      IF(I.NE.IP) WRITE(6,155) IP
  155 FORMAT(/, ' NODE NO.', I5,' IN DATA FILE IS MISSING')
      IF(I.NE.IP) STOP

  150 CONTINUE

C--------------------------------------
C     READ ELEMENT NODAL CONNECTION
C--------------------------------------
      READ(7,1) TEXT

      DO 160 IE=1,NELEM

      READ(7,*)  I, (INTMAT(I,J), J=1,3)

      IBCE(I) = 0

      IF(I.NE.IE) WRITE(6,165) IE
  165 FORMAT(/, ' ELEMENT NO.', I5, ' IN DATA FILE IS MISSING')
      IF(I.NE.IE) STOP

  160 CONTINUE

C--------------------------------------
C     READ INFLOW BOUNDARY ELEMENT
C--------------------------------------
      READ(7,1) TEXT

      DO 170 IE=1,NBOU
      READ(7,*)  I, (INTBOU(I,J), J=1,2)
      IF(I.NE.0) IBCE(I) = 1
  170 CONTINUE

C-----------------------------
C     PRINT OUT TITLE
C-----------------------------
      WRITE(6,200) NPOI, NELEM, NITER, TOL
  200 FORMAT(/,'   THE FINITE ELEMENT MODEL CONSISTS OF :',/,
     *         '      NUMBER OF NODES              =', I6, /,
     *         '      NUMBER OF ELEMENTS           =', I6, /,
     *         '      NUMBER OF MAX. ITERATION     =', I6, /,
     *         '      SPECIFIED STOPPING TOLERANCE =', E10.5 ,/)

C-------------------------------------------------------------------C

C-----------------------------
C     START CALCULATION
C-----------------------------
      DO 500 ITER=1,NITER

      NEQ = NPOI

      DO 30 I=1,NPOI
      AREAVG(I) = 0.
   30 CONTINUE

C-----------------------------------------------
C     CHECK ELEMENT THAT DON'T HAVE DOWNWIND NODE
C-----------------------------------------------

      CALL CHK(NELEM, COORD, INTMAT,  MXPOI, IBCU,
     *          IBCV, MXELE, AREAVG,      U,    V)
```

```
C---------------------------------
C     SOLVE FOR U & V -VELOCITIES
C---------------------------------

      CALL SOLVEUV( NPOI, NELEM,   NEQ,  DEN,    VIS,  COORD, INTMAT,
     *              MXPOI, MXELE, MXNEQ, ASYS,      U,      V,      P,
     *               IBCU,  IBCV,    RU,   RV, AREAVG )

C---------------------------------
C     CONSTRUCT UHAT, VHAT & Kp
C---------------------------------

      CALL COEFF(NPOI, NELEM, COORD, INTMAT, MXPOI, MXELE,
     "           UHAT,  VHAT,   CKP,   ASYS, U,     V,
     "             RU,    RV)

C---------------------------------
C     SOLVE FOR PRESSURE
C---------------------------------

      CALL SOLVEP( NPOI, NELEM,   NEQ,  CKP, COORD, INTMAT, INTBOU,
     *             MXPOI, MXELE, MXNEQ, UHAT,  VHAT,      P,   IBCP,
     *              POLD,  IBCE,     U,    V)

C---------------------------------
C     UPDATE THE VELOCITIES
C---------------------------------

      CALL UPDATE( NPOI, NELEM, COORD, INTMAT, UHAT, VHAT,
     *             MXPOI, MXELE, MXNEQ,   ASYS,    U,    V,
     *                 P,  IBCU,  IBCV,   UOLD, VOLD )

C---------------------------------
C     CHECK FOR CONVERGENCE
C---------------------------------
      SUMU = 0.
      SUMV = 0.
      SUMP = 0.

      SUMDU = 0.
      SUMDV = 0.
      SUMDP = 0.

      DO 900 I=1,NPOI

      SUMDU = SUMDU + ABS(UOLD(I)-U(I))
      SUMU  = SUMU  + ABS(U(I))

      SUMDV = SUMDV + ABS(VOLD(I)-V(I))
      SUMV  = SUMV  + ABS(V(I))

      SUMDP = SUMDP + ABS(POLD(I)-P(I))
      SUMP  = SUMP  + ABS(P(I))

  900 CONTINUE

      ERRORU = SUMDU/SUMU
      ERRORV = SUMDV/SUMV
      ERRORP = SUMDP/SUMP

      OPEN(UNIT=11, FILE='ERROR', STATUS='UNKNOWN')

      WRITE(6,910)  ITER, ERRORU, ERRORV, ERRORP
      WRITE(11,910) ITER, ERRORU, ERRORV, ERRORP
  910 FORMAT(3X, 'ITER =  ', I5, 3X, E15.10, 3X,
     *       E15.10, 3X, E15.10                  )

C---------------------------------
C     UNDER RELAXATION
C---------------------------------

      DO 350 I=1,NPOI
      U(I) = 0.5*UOLD(I)+0.5*U(I)
      V(I) = 0.5*VOLD(I)+0.5*V(I)
      P(I) = 0.5*POLD(I)+0.5*P(I)
  350 CONTINUE
```

```
C-------------------------------------------------------
C
C     WRITE THE OUTPUT FILE IF THE SOLUTION CONVERGE
C
C-------------------------------------------------------

      IF(ERRORU.LE.TOL.OR.ERRORV.LE.TOL.OR.ERRORP.LE.TOL
     *   .OR.ITER.EQ.NITER) THEN

C-------------------------------------------------------
      CALL GETTIM(enhour, enminute, ensecond, enhund)
C-------------------------------------------------------

C-------------------------------------------------------------
      WRITE(6,300) sthour, stminute, stsecond, sthund
  300 FORMAT('  Start time :',3X,I2,':',I2.2,':',I2.2,':',I2.2)
      WRITE(6,310) enhour, enminute, ensecond, enhund
  310 FORMAT('    End time :',3X,I2,':',I2.2,':',I2.2,':',I2.2)
C-------------------------------------------------------------

  930 WRITE(6,940)
  940 FORMAT(/, ' ENTER THE OUTPUT FILE NAME',/)
      READ(5 , '(A)', ERR=930 ) NAME2

      OPEN(UNIT=8, FILE=NAME2 , STATUS='NEW',ERR=930)

      WRITE(8,942)
  942 FORMAT('1 MSC/NASTRAN PAGE',/)
      WRITE(8,944)
  944 FORMAT('0')
      WRITE(8,946)
  946 FORMAT('  D I S P L A C E M E N T')

      AAA = 0.

      DO 1000 I=1,NPOI
      WRITE(8,950) I, U(I), V(I), AAA, P(I), AAA, AAA
  950 FORMAT(I6,2X,'G', 6E12.4)
 1000 CONTINUE

      WRITE(8,948)
  948 FORMAT('0')

C---------------------------------------
C     CREATE DATA FOR VELOCITY REMESH
C---------------------------------------

 1100 WRITE(6,1200)
 1200 FORMAT(/, ' ENTER THE FILE NAME FOR REMESHING',/)
      READ(5 , '(A)', ERR=930 ) NAME3
      OPEN(UNIT=12, FILE=NAME3 , STATUS='NEW',ERR=930)

      DO 1300 I=1,NPOI
      VSUM(I) = SQRT(U(I)*U(I) + V(I)*V(I))
 1300 CONTINUE

      WRITE(12,1350) NPOI
 1350 FORMAT(I5)

      DO 1400 I=1,NPOI
      WRITE(12,1450) I,VSUM(I)
 1450 FORMAT(I5,2X,E12.5)
 1400 CONTINUE

      ENDIF

C-------------------------------------------------------------------
      IF(ERRORU.LE.TOL.OR.ERRORV.LE.TOL.OR.ERRORP.LE.TOL
     *   .OR.ITER.EQ.NITER) STOP
C-------------------------------------------------------------------

  500 CONTINUE

      STOP
      END

C-------------------------------------------------------------------C
```

```
C
C     SUBROUTINE FOR SOLVE U & V -VELOCITIES
C

      SUBROUTINE SOLVEUV( NPOI, NELEM,   NEQ,  DEN, VIS, COORD, INTMAT,
     *                   MXPOI, MXELE, MXNEQ, ASYS, U,   V,     P,
     *                    IBCU,  IBCV,    RU,   RV, AREAVG)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), SYSK(MXNEQ,MXNEQ), ASYS(MXNEQ,MXNEQ)
      DIMENSION SYSRX(MXNEQ), SYSRY(MXNEQ), SOL(MXNEQ), AREAVG(MXPOI)
      DIMENSION U(MXPOI), V(MXPOI), P(MXPOI), RU(MXPOI), RV(MXPOI)
      DIMENSION SYS(MXNEQ),ADIF(3,3), AELE(3,3), ACOV(3,3), RPX(3), RPY(3)

      INTEGER  INTMAT(MXELE,3)
      INTEGER  IBCU(MXPOI), IBCV(MXPOI)

C-----------------------------------------
C     RESET THE SYSTEM OF EQUATION
C-----------------------------------------
      DO 666 I=1,NPOI
      SYSRX(I) = 0.
      SYSRY(I) = 0.
      DO 666 J=1,NPOI
      SYSK(I,J) = 0.
  666 CONTINUE

C-----------------------------------------
C     LOOP OVER THE NUMBER OF ELEMENTS
C-----------------------------------------
      DO 500 IE=1,NELEM

C-----------------------------------------
C     FIND ELEMENT LOCAL COORDINATES :
C-----------------------------------------
      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)

      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'         )
      IF(AREA.LE.0.) STOP

      B1 = YG2 - YG3
      B2 = YG3 - YG1
      B3 = YG1 - YG2

      C1 = XG3 - XG2
      C2 = XG1 - XG3
      C3 = XG2 - XG1

C=========================================================
C     SET UP [A] METRIX : (FROM THE DIFFUSION TERM)
C=========================================================
      ADIF(1,1) = ((B1*B1)+(C1*C1))/(4.*AREA)
      ADIF(1,2) = ((B1*B2)+(C1*C2))/(4.*AREA)
      ADIF(1,3) = ((B1*B3)+(C1*C3))/(4.*AREA)
      ADIF(2,2) = ((B2*B2)+(C2*C2))/(4.*AREA)
      ADIF(2,3) = ((B2*B3)+(C2*C3))/(4.*AREA)
      ADIF(3,3) = ((B3*B3)+(C3*C3))/(4.*AREA)
      ADIF(2,1) = ADIF(1,2)
      ADIF(3,1) = ADIF(1,3)
      ADIF(3,2) = ADIF(2,3)
```

```
C=========================================================
C     SET UP [A] METRIX : (FROM THE CONVECTION TERM)
C=========================================================

      CALL STREAM(   IE, INTMAT, COORD,      U,  V,  MXPOI,
     *            MXELE,   ACOV,   DEN, AREAVG)

C=================================
C     SET UP [RPX] METRIX :
C=================================
      RPX(1) = -(B1*P(II)+B2*P(JJ)+B3*P(KK))/6.
      RPX(2) = RPX(1)
      RPX(3) = RPX(1)

C=================================
C     SET UP [RPY] METRIX :
C=================================
      RPY(1) = -(C1*P(II)+C2*P(JJ)+C3*P(KK))/6.
      RPY(2) = RPY(1)
      RPY(3) = RPY(1)

C=============================================
C     SET UP [AELE] MATRIX : [ADIF]+[ACOV]
C            [RELE] MATRIX : [RPX]
C=============================================
      DO 30 I=1,3
      DO 30 J=1,3
      AELE(I,J) = ACOV(I,J) + VIS*ADIF(I,J)
   30 CONTINUE

C=========================================
C     ASSEMBLE THESE ELEMENT EQUATIONS
C     INTO THE SYSTEM EQUATIONS
C=========================================

      CALL ASSMBLE(  IE, INTMAT,  AELE,   RPX,   RPY,
     *             SYSK,  SYSRX, SYSRY, MXNEQ, MXELE)

  500 CONTINUE

C-------------------------------------------------------------------C

C=================================================
C     SOLVE U-VELOCITY
C=================================================
      DO 510 I=1,NPOI
      SYS(I) = SYSRX(I)
      DO 510 J=1,NPOI
      ASYS(I,J) = SYSK(I,J)
  510 CONTINUE

C==================================
C     APPLY BOUNDARY CONDITION
C==================================

      CALL APPLYBC(NPOI, IBCU, U, SYSK, SYSRX, MXPOI, MXNEQ )

C==================================
C     SOLVE SYSTEM OF EQUATION
C==================================

      CALL GS(SYSK, SYSRX, SOL, NEQ, MXNEQ, U)

C-------------------------------------------------------------------C

      DO 100 I=1,NPOI
      U(I)  = SOL(I)
  100 CONTINUE
      DO 110 I=1,NPOI
      SUM = 0.
      DO 120 J=1,NPOI
      SUM = SUM + ASYS(I,J)*U(J)
  120 CONTINUE
      RU(I) = SUM - SYS(I)
  110 CONTINUE

C-------------------------------------------------------------------C
```

```
C=================================================
C     SOLVE V-VELOCITY
C=================================================
      DO 520 I=1,NPOI
      SYS(I) = SYSRY(I)
      DO 520 J=1,NPOI
      SYSK(I,J) = ASYS(I,J)
  520 CONTINUE

C===================================
C     APPLY BOUNDARY CONDITION
C===================================

      CALL APPLYBC(NPOI, IBCV, V, SYSK, SYSRY, MXPOI, MXNEQ )

C===================================
C     SOLVE SYSTEM OF EQUATION
C===================================

      CALL GS(SYSK, SYSRY, SOL, NEQ, MXNEQ, V)

C-------------------------------------------------------------------C

      DO 200 I=1,NPOI
      V(I)  = SOL(I)
  200 CONTINUE

      DO 210 I=1,NPOI
      SUM = 0.
      DO 220 J=1,NPOI
      SUM = SUM + ASYS(I,J)*V(J)
  220 CONTINUE
      RV(I) = SUM - SYS(I)
  210 CONTINUE

C-------------------------------------------------------------------C

      RETURN
      END

C===================================================================C

C
C   SUBROUTINE FOR SET UP COEFFICIENT
C

      SUBROUTINE COEFF(NPOI, NELEM, COORD, INTMAT, MXPOI, MXELE,
     *                 UHAT, VHAT,  CKP,   ASYS,   U, V, RU, RV)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2)
      DIMENSION UHAT(MXPOI), VHAT(MXPOI), CKP(MXPOI),RU(MXPOI)
      DIMENSION ASYS(MXPOI,MXPOI),U(MXPOI),V(MXPOI), RV(MXPOI)
      INTEGER  INTMAT(MXELE,3)

      DO 100 I=1,NPOI
      CKP(I) = 0.
  100 CONTINUE

C
C     LOOP OVER THE NUMBER OF ELEMENTS :
C

      DO 500 IE=1,NELEM

C
C     FIND ELEMENT LOCAL COORDINATES :
C

      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)
```

```
      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'          )
      IF(AREA.LE.0.) STOP

C=======================
C     SET UP CKP  :
C=======================

      CKP(II) = CKP(II) + (AREA/(3.*ASYS(II,II)))
      CKP(JJ) = CKP(JJ) + (AREA/(3.*ASYS(JJ,JJ)))
      CKP(KK) = CKP(KK) + (AREA/(3.*ASYS(KK,KK)))

  500 CONTINUE

C============================
C     SET UP UHAT & VHAT
C============================

      DO 600 I=1,NPOI
      SUM1 = 0.
      SUM2 = 0.
      DO 610 J=1,NPOI
      IF(J.NE.I) THEN
      SUM1 = SUM1 + (-ASYS(I,J)*U(J)/ASYS(I,I))
      SUM2 = SUM2 + (-ASYS(I,J)*V(J)/ASYS(I,I))
      ENDIF
  610 CONTINUE

      UHAT(I) = SUM1 + (RU(I)/ASYS(I,I))
      VHAT(I) = SUM2 + (RV(I)/ASYS(I,I))

  600 CONTINUE

C-------------------------------------------------------------------C

      RETURN
      END

C===================================================================C

C
C   SUBROUTINE FOR SOLVE PRESSURE
C

      SUBROUTINE SOLVEP(NPOI,   NELEM,  NEQ,   CKP,   COORD,
     *                  INTMAT, INTBOU, MXPOI, MXELE, MXNEQ,
     *                  UHAT,   VHAT,   P,     IBCP,  POLD,
     *                  IBCE,   U,      V)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), SYSK(MXNEQ,MXNEQ)
      DIMENSION SYSR(MXNEQ), SOL(MXNEQ),  POLD(MXPOI)
      DIMENSION UHAT(MXPOI), VHAT(MXPOI), CKP(MXPOI)
      DIMENSION RPU(3),     RPV(3),  P(MXPOI), RB(3)
      DIMENSION U(MXPOI),   V(MXPOI)
      DIMENSION AKELE(3,3), RELE(3), ADIF(3,3)

      INTEGER  INTMAT(MXELE,3), INTBOU(MXELE,2)
      INTEGER  IBCP(MXPOI), IBCE(MXELE)

      DO 666 I=1,NPOI
      SYSR(I) = 0.
      POLD(I) = P(I)
      DO 666 J=1,NPOI
      SYSK(I,J) = 0.
  666 CONTINUE
```

```
C
C     LOOP OVER THE NUMBER OF ELEMENTS :
C

      DO 500 IE=1,NELEM

C
C     FIND ELEMENT LOCAL COORDINATES :
C

      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)

      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'         )
      IF(AREA.LE.0.) STOP

      B1 = YG2 - YG3
      B2 = YG3 - YG1
      B3 = YG1 - YG2

      C1 = XG3 - XG2
      C2 = XG1 - XG3
      C3 = XG2 - XG1

C================================
C     SET UP [A1] METRICES :
C================================
      DO 30 I = 1,3
      RB(I) = 0.
      DO 30 J = 1,3
      ADIF(I,J) = 0.
   30 CONTINUE

      ADIF(1,1) = ((B1*B1)+(C1*C1))/(4.*AREA)
      ADIF(1,2) = ((B1*B2)+(C1*C2))/(4.*AREA)
      ADIF(1,3) = ((B1*B3)+(C1*C3))/(4.*AREA)
      ADIF(2,2) = ((B2*B2)+(C2*C2))/(4.*AREA)
      ADIF(2,3) = ((B2*B3)+(C2*C3))/(4.*AREA)
      ADIF(3,3) = ((B3*B3)+(C3*C3))/(4.*AREA)
      ADIF(2,1) = ADIF(1,2)
      ADIF(3,1) = ADIF(1,3)
      ADIF(3,2) = ADIF(2,3)

C============================
C    SET UP [RU] MATRIX
C============================

      CRU   = (UHAT(II)+UHAT(JJ)+UHAT(KK))/6.
      RPU(1) = CRU*B1
      RPU(2) = CRU*B2
      RPU(3) = CRU*B3

C============================
C    SET UP [RV] MATRIX
C============================

      CRV   = (VHAT(II)+VHAT(JJ)+VHAT(KK))/6.
      RPV(1) = CRV*C1
      RPV(2) = CRV*C2
      RPV(3) = CRV*C3
```

```
C=========================
C    SET UP [RB] MATRIX
C=========================

      IF(IBCE(IE).EQ.1) THEN
      IB = INTBOU(IE,1)
      JB = INTBOU(IE,2)

      XB1 = COORD(IB,1)
      XB2 = COORD(JB,1)

      YB1 = COORD(IB,2)
      YB2 = COORD(JB,2)

      UNX = (YB2 - YB1)
      UNY = (XB1 - XB2)

      UAVG = (U(IB) + U(JB))/2.
      VAVG = (V(IB) + V(JB))/2.

      IF(IB.EQ.II) THEN
      RB(1) = (UAVG*UNX + VAVG*UNY)/2.
      RB(2) = (UAVG*UNX + VAVG*UNY)/2.
      RB(3) = 0.
      ENDIF

      IF(IB.EQ.JJ) THEN
      RB(1) = 0.
      RB(2) = (UAVG*UNX + VAVG*UNY)/2.
      RB(3) = (UAVG*UNX + VAVG*UNY)/2.
      ENDIF

      IF(IB.EQ.KK) THEN
      RB(1) = (UAVG*UNX + VAVG*UNY)/2.
      RB(2) = 0.
      RB(3) = (UAVG*UNX + VAVG*UNY)/2.
      ENDIF

      ENDIF

C============================
C    SET UP [KX+KY] MATRIX
C============================

      COEFP = (CKP(II)+CKP(JJ)+CKP(KK))/3.

      DO 120 I=1,3
      DO 120 J=1,3
      AKELE(I,J) = 0.
      AKELE(I,J) = COEFP*ADIF(I,J)
  120 CONTINUE

C==================================
C    SET UP [RELE] : [RU]+[[RV]
C==================================

      DO 140 I=1,3
      RELE(I) = 0.
      RELE(I) = RPU(I)+RPV(I)-RB(I)
  140 CONTINUE

C=======================================
C    AEEEMBLE THESE ELEMENT EQUATIONS
C    INTO THE SYSTEM EQUATIONS
C=======================================

      CALL ASSMP(IE, INTMAT, AKELE, RELE, SYSK, SYSR,
     *           MXNEQ, MXELE   )

C-------------------------------------------------------------------C

  500 CONTINUE
```

```
C==================================
C     APPLY BOUNDARY CONDITION
C==================================

      CALL APPLYBC(NPOI,  IBCP, P,
     *             SYSK,  SYSR, MXPOI, MXNEQ )

C==================================
C     SOLVE SYSTEM OF EQUATION
C==================================
      TOL = 1.e-7
      MAXIT = 1000

      CALL PCG(SYSK,SYSR,SOL,TOL,MAXIT,MXNEQ,NEQ,POLD)

C-------------------------------------------------------------------C

      DO 200 I=1,NPOI
      P(I) = SOL(I)
      IF(ABS(P(I)).LT.1.E-10) THEN
      P(I) = 0.
      ENDIF
  200 CONTINUE

C-------------------------------------------------------------------C

      RETURN
      END

C===================================================================C

C
C     SUBROUTINE FOR UPDATE VELOCITIES
C

      SUBROUTINE UPDATE(NPOI,  NELEM, COORD, INTMAT, UHAT, VHAT,
     *                  MXPOI, MXELE, MXNEQ, ASYS,   U,    V,
     *                  P,     IBCU,  IBCV,  UOLD,   VOLD )


      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), ASYS(MXNEQ,MXNEQ)
      DIMENSION SYSRY(MXNEQ), U(MXPOI), P(MXPOI), UHAT(MXPOI)
      DIMENSION SYSRX(MXNEQ), V(MXPOI), VHAT(MXPOI)
      DIMENSION RPX(3), RPY(3), UOLD(MXPOI), VOLD(MXPOI)

      INTEGER  INTMAT(MXELE,3)
      INTEGER  IBCU(MXPOI), IBCV(MXPOI)

C-----------------------------------
C     RESET THE SYSTEM OF EQUATION
C     AND COLLECT THE OLD VALUE
C-----------------------------------
      DO 666 I=1,NPOI
      SYSRX(I) = 0.
      SYSRY(I) = 0.
      UOLD(I) = U(I)
      VOLD(I) = V(I)
  666 CONTINUE

C-----------------------------------------
C     LOOP OVER THE NUMBER OF ELEMENTS
C-----------------------------------------
      DO 500 IE=1,NELEM

C-----------------------------------------
C     FIND ELEMENT LOCAL COORDINATES :
C-----------------------------------------

      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)
```

```
      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      B1 = YG2 - YG3
      B2 = YG3 - YG1
      B3 = YG1 - YG2

      C1 = XG3 - XG2
      C2 = XG1 - XG3
      C3 = XG2 - XG1

C=================================
C     SET UP [RPX] METRIX :
C=================================
      RPX(1) = -(B1*P(II)+B2*P(JJ)+B3*P(KK))/6.
      RPX(2) = RPX(1)
      RPX(3) = RPX(1)

C=================================
C     SET UP [RPY] METRIX :
C=================================
      RPY(1) = -(C1*P(II)+C2*P(JJ)+C3*P(KK))/6.
      RPY(2) = RPY(1)
      RPY(3) = RPY(1)

C=============================================
C     ASSEMBLE THESE ELEMENT EQUATIONS
C     INTO THE SYSTEM EQUATIONS
C=============================================

      CALL ASSMUP(IE,    INTMAT, RPX,   RPY,
     *            SYSRX, SYSRY,  MXNEQ, MXELE )

C-------------------------------------------------------------------C

  500 CONTINUE

C=============================================
C     CALCULATE THE MATRIX SYSR(I)
C=============================================
      DO 510 I=1,NPOI
      SYSRX(I) = SYSRX(I)/ASYS(I,I)
      SYSRY(I) = SYSRY(I)/ASYS(I,I)
  510 CONTINUE

C=============================================
C    EVALUATE THE NEW VALUE OF VELOCITIES
C=============================================

      DO 600 I=1,NPOI

      IF(IBCU(I).EQ.1) THEN
      U(I) = UOLD(I)
      ELSE
      U(I) = UHAT(I) + SYSRX(I)
      ENDIF
      IF(ABS(U(I)).LE.1.E-10) THEN
      U(I) = 0.
      ENDIF

      IF(IBCV(I).EQ.1) THEN
      V(I) = VOLD(I)
      ELSE
      V(I) = VHAT(I) + SYSRY(I)
      ENDIF
      IF(ABS(V(I)).LE.1.E-10) THEN
      V(I) = 0.
      ENDIF

  600 CONTINUE

C-------------------------------------------------------------------C

      RETURN
      END
```

```
      SUBROUTINE CHK(NELEM, COORD, INTMAT,  MXPOI, IBCU,
     *                IBCV, MXELE, AREAVG,      U,    V)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2)
      DIMENSION U(MXPOI), V(MXPOI), AREAVG(MXPOI)

      INTEGER   INTMAT(MXELE,3), IBCU(MXPOI), IBCV(MXPOI)

C
C     LOOP OVER THE NUMBER OF ELEMENTS :
C

      DO 500 IE=1,NELEM

      CHECK3 = 0.

C
C     FIND ELEMENT LOCAL COORDINATES :
C

      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)

      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'          )
      IF(AREA.LE.0.) STOP

C=============================================================
C     FIND MASS FLOW RATE PAST THE SIDE OF ELEMENT
C
C     F1 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 1
C     F2 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 2
C     F3 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 3
C
C=============================================================

      F1 = -(XG3-XG2)*(0.5*(V(JJ)+V(KK)))+(YG3-YG2)*(0.5*(U(JJ)+U(KK)))
      F2 = -(XG1-XG3)*(0.5*(V(KK)+V(II)))+(YG1-YG3)*(0.5*(U(KK)+U(II)))
      F3 = -(XG2-XG1)*(0.5*(V(II)+V(JJ)))+(YG2-YG1)*(0.5*(U(II)+U(JJ)))

C-------------------------
C     CHECK FOR NODE 1
C-------------------------

      UJ   =  U(II)
      VJ   =  V(II)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))
        CHECK2 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))

          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF

      ENDIF
```

```
C-------------------------
C     CHECK FOR NODE 2
C-------------------------

      UJ   =  U(JJ)
      VJ   =  V(JJ)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))
        CHECK2 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))

          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF

      ENDIF

C-------------------------
C     CHECK FOR NODE 3
C-------------------------

      UJ   =  U(KK)
      VJ   =  V(KK)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))
        CHECK2 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))

          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF

      ENDIF

C-----------------------------------------------------------------
      IF(CHECK3.NE.1.) THEN

      DIVIDE = 3.

      DO 10 I=1,3
      NN = INTMAT(IE,I)
      IF(IBCU(NN).EQ.1.AND.IBCV(NN).EQ.1.AND.
     *      U(NN).EQ.0..AND.V(NN).EQ.0.) THEN
      DIVIDE = DIVIDE - 1.
      ENDIF
   10 CONTINUE

      IF(DIVIDE.NE.0.) THEN
      DO 15 I=1,3
      NN = INTMAT(IE,I)
      IF(IBCU(NN).EQ.1.AND.IBCV(NN).EQ.1.AND.
     *      U(NN).EQ.0..AND.V(NN).EQ.0.) THEN
      AREAVG(NN) = 0.
      ELSE
      AREAVG(NN) = AREAVG(NN) + (AREA/DIVIDE)
      ENDIF
   15 CONTINUE
      ENDIF

      ENDIF
C-----------------------------------------------------------------

  500 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------

C
C    SUBROUTINE FOR STREAMLINE-UPWIND
C
      SUBROUTINE STREAM(IE,    INTMAT, COORD, U,     V,     MXPOI,
     *                  MXELE, ACOV,   DEN, AREAVG)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), U(MXPOI), V(MXPOI)
      DIMENSION ACOV(3,3), AREAVG(MXPOI)
```

```
      INTEGER INTMAT(MXELE,3)

C
C     FIND ELEMENT LOCAL COORDINATES :
C
      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)

      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'         )
      IF(AREA.LE.0.) STOP

C==============================
C     SET UP [ADIF] METRIX :
C==============================
      DO 30 I = 1,3
      DO 30 J = 1,3
      ACOV(I,J) = 0.
   30 CONTINUE

C==============================
C     SET UP [A] METRIX :
C==============================

C=====================================================================
C  FIND UNIT VECTOR AT EACH SIDE OF ELEMENT
C
C  UNX1,UNY1 = Unit vector point out on the opposite side of node 1
C  UNX2,UNY2 = Unit vector point out on the opposite side of node 2
C  UNX3,UNY3 = Unit vector point out on the opposite side of node 3
C=====================================================================

      UNX1 = (YG3-YG2)
      UNY1 = (XG2-XG3)

      UNX2 = (YG1-YG3)
      UNY2 = (XG3-XG1)

      UNX3 = (YG2-YG1)
      UNY3 = (XG1-XG2)

C=========================================================
C     FIND MASS FLOW RATE PAST THE SIDE OF ELEMENT
C
C       F1 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 1
C       F2 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 2
C       F3 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 3
C=========================================================

      F1 = (0.5*(U(JJ)+U(KK))*UNX1)+(0.5*(V(JJ)+V(KK))*UNY1)
      F2 = (0.5*(U(II)+U(KK))*UNX2)+(0.5*(V(II)+V(KK))*UNY2)
      F3 = (0.5*(U(II)+U(JJ))*UNX3)+(0.5*(V(II)+V(JJ))*UNY3)

C=====================================
C     FIND STREAMLINE UPWIND NODE
C     SET UP [ACOV] METRIX :
C=====================================

      COEF = 0.
```

```
C-------------------------
C     CHECK FOR NODE 1
C-------------------------

      UJ   =  U(II)
      VJ   =  V(II)

      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))
      CHECK2 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))

      IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

      IF(F1.LT.0.) THEN

      IF(F2.LT.0..AND.F3.GT.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F3.LT.0..AND.F2.GT.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F2.GT.0..AND.F3.EQ.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F3.GT.0..AND.F2.EQ.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F2.EQ.F3) THEN
        FP = 0.5
        FN = 0.5
      ELSEIF(ABS(F2).GE.ABS(F1)) THEN
        FP = 1.
        FN = 0.
      ELSEIF(ABS(F3).GE.ABS(F1)) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F2.GT.0..AND.F3.GT.0.) THEN

        IF(U(JJ).EQ.0..AND.V(JJ).EQ.0.) THEN
           FP = 0.
           FN = 1.
        ELSEIF(U(KK).EQ.0..AND.V(KK).EQ.0.) THEN
           FP = 1.
           FN = 0.
        ELSE
           FP = ABS(F2/F1)
           FN = 1.-FP
        ENDIF

      ENDIF

      X = FP*XG2 + FN*XG3
      Y = FP*YG2 + FN*YG3

      US   = SQRT((UJ*UJ)+(VJ*VJ))

      DS   = SQRT((X-XG1)*(X-XG1)+(Y-YG1)*(Y-YG1))

      AREA = AREA + AREAVG(II)

      COEF = (DEN*US*AREA)/DS

      ACOV(1,1) =     COEF
      ACOV(1,2) = -FP*COEF
      ACOV(1,3) = -FN*COEF

      ENDIF
      ENDIF
      ENDIF
```

```
C--------------------------
C     CHECK FOR NODE 2
C--------------------------

      UJ   =  U(JJ)
      VJ   =  V(JJ)

      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))
      CHECK2 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))

      IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

      IF(F2.LT.0.) THEN

      IF(F3.LT.0..AND.F1.GT.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F1.LT.0..AND.F3.GT.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F3.GT.0..AND.F1.EQ.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F1.GT.0..AND.F3.EQ.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F1.EQ.F3) THEN
        FP = 0.5
        FN = 0.5
      ELSEIF(ABS(F1).GE.ABS(F2)) THEN
        FP = 0.
        FN = 1.
      ELSEIF(ABS(F3).GE.ABS(F2)) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F1.GT.0..AND.F3.GT.0.) THEN

        IF(U(II).EQ.0..AND.V(II).EQ.0.) THEN
           FP = 1.
           FN = 0.
        ELSEIF(U(KK).EQ.0..AND.V(KK).EQ.0.) THEN
           FP = 0.
           FN = 1.
        ELSE
           FP = ABS(F3/F2)
           FN = 1.-FP
        ENDIF

      ENDIF

      X = FP*XG3 + FN*XG1
      Y = FP*YG3 + FN*YG1

      US   = SQRT((UJ*UJ)+(VJ*VJ))

      DS   = SQRT((X-XG2)*(X-XG2)+(Y-YG2)*(Y-YG2))

      AREA = AREA + AREAVG(JJ)

      COEF = (DEN*US*AREA)/DS

      ACOV(2,1) = -FN*COEF
      ACOV(2,2) =     COEF
      ACOV(2,3) = -FP*COEF

      ENDIF
      ENDIF
      ENDIF
```

```
C--------------------------
C     CHECK FOR NODE 3
C--------------------------

      UJ    =  U(KK)
      VJ    =  V(KK)

      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))
      CHECK2 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))

      IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

      IF(F3.LT.0.) THEN

      IF(F1.LT.0..AND.F2.GT.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F2.LT.0..AND.F1.GT.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F1.GT.0..AND.F2.EQ.0.) THEN
        FP = 1.
        FN = 0.
      ELSEIF(F2.GT.0..AND.F1.EQ.0.) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F1.EQ.F2) THEN
        FP = 0.5
        FN = 0.5
      ELSEIF(ABS(F1).GE.ABS(F3)) THEN
        FP = 1.
        FN = 0.
      ELSEIF(ABS(F2).GE.ABS(F3)) THEN
        FP = 0.
        FN = 1.
      ELSEIF(F1.GT.0..AND.F2.GT.0.) THEN

        IF(U(II).EQ.0..AND.V(II).EQ.0.) THEN
           FP = 0.
           FN = 1.
        ELSEIF(U(JJ).EQ.0..AND.V(JJ).EQ.0.) THEN
           FP = 1.
           FN = 0.
        ELSE
           FP = ABS(F1/F3)
           FN = 1.-FP
        ENDIF

      ENDIF

      X = FN*XG2 + FP*XG1
      Y = FN*YG2 + FP*YG1

      US    = SQRT((UJ*UJ)+(VJ*VJ))

      DS    = SQRT((X-XG3)*(X-XG3)+(Y-YG3)*(Y-YG3))

      AREA = AREA + AREAVG(KK)

      COEF = (DEN*US*AREA)/DS

      ACOV(3,1) = -FP*COEF
      ACOV(3,2) = -FN*COEF
      ACOV(3,3) =     COEF

      ENDIF
      ENDIF
      ENDIF

      RETURN
      END

C=============================================================C
```

```
C
C     SUBROUTINE FOR APPLY BOUNDARY CONDITION
C
      SUBROUTINE APPLYBC(NPOI,  IBC,  UVP,
     *                   SYSK, SYSR, MXPOI, MXNEQ )
C
C     APPLY BOUNDARY CONDITIONS BEFORW SOLVING FOR NODAL
C     WITH CONDITION CODES OF:
C        0 = FREE TO CHANGE
C        1 = FIXED AS SPECIFIED
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION SYSK(MXNEQ,MXNEQ), SYSR(MXNEQ)
      DIMENSION UVP(MXNEQ)

      INTEGER IBC(MXPOI)

C---------------------------------------------
C     APPLY BOUNDARY CONDITIONS FOR NODAL
C---------------------------------------------
      DO 100 IEQ=1,NPOI

      IF(IBC(IEQ).EQ.0) GOTO 100

      DO 110 IR=1,NPOI
      IF(IR.EQ.IEQ) GOTO 110
      SYSR(IR) = SYSR(IR) - SYSK(IR,IEQ)*UVP(IEQ)
      SYSK(IR,IEQ) = 0.
  110 CONTINUE

      DO 120 IC=1,NPOI
      SYSK(IEQ,IC) = 0.
  120 CONTINUE

      SYSK(IEQ,IEQ) = 1.
      SYSR(IEQ) = UVP(IEQ)

  100 CONTINUE

      RETURN
      END

C-------------------------------------------------------------

C
C     SUBROUTINE GAUSS-SEIDEL FOR SOLVING THE SYSTEM OF EQUATIONS
C

      SUBROUTINE GS(A,B,X,N,MXNEQ,UVP)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ), X(MXNEQ), UVP(MXNEQ)

      COUNT = 0
      MAXIT = 1

      DO 25 I=1,N
      X(I) =   UVP(I)
   25 CONTINUE

   30 IF(COUNT.LT.MAXIT) THEN
      DO 40 I=1,N
      SUM = 0.
      DO 35 J=1,N
      IF(J.NE.I) SUM = SUM + A(I,J)*X(J)
   35 CONTINUE
      XNEW = (B(I) - SUM)/A(I,I)
      X(I) = XNEW
   40 CONTINUE
      COUNT = COUNT + 1
      GOTO 30
      ENDIF

      RETURN
      END

C-----------------------------------------------
```

```
C
C    SUBROUTINE PRECONDITIONED CONJUGATE GRADIENT METHOD
C

      SUBROUTINE PCG(A,B,X,TOL,MAXIT,MXNEQ,N,XOLD)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ), X(MXNEQ), Q(MXNEQ)
      DIMENSION R(MXNEQ), D(MXNEQ), P(MXNEQ), S(MXNEQ), XOLD(MXNEQ)

      ITER = 1

C-----------------------------------------------------------------------
C     CONSTRUCT PERCONDITION MATRIX P (BUT THIS WILL FIND P Inverse)
C-----------------------------------------------------------------------
      DO 200 I=1,N
      X(I) = XOLD(I)
      P(I) = 0.
  200 CONTINUE

      DO 210 I=1,N
      P(I) = 1./A(I,I)
  210 CONTINUE

C--------------------
C     FIND Ro
C--------------------
      DO 10 I=1,N
      SUM = 0.
      DO 20 J=1,N
      SUM = SUM + A(I,J)*X(J)
   20 CONTINUE
      R(I) = B(I) - SUM
   10 CONTINUE

C--------------------
C     FIND Do
C--------------------
      DO 30 I=1,N
      D(I) = 0.
      D(I) = R(I)*P(I)
   30 CONTINUE

C--------------------
C     FIND D-New
C--------------------
      DNEW = 0.
      DO 40 I=1,N
      DNEW = DNEW + R(I)*D(I)
   40 CONTINUE

      DEL = DNEW

    5 IF(ITER.LT.MAXIT.AND.DNEW.GT.(TOL*TOL*DEL)) THEN

C--------------------
C     FIND Q(i+1)
C--------------------
      DO 50 I=1,N
      SUM = 0.
      DO 60 J=1,N
      SUM = SUM + A(I,J)*D(J)
   60 CONTINUE
      Q(I) = SUM
   50 CONTINUE

C--------------------
C     FIND ALPHA
C--------------------
      BOTTOM = 0.
      DO 70 I=1,N
      BOTTOM = BOTTOM + D(I)*Q(I)
   70 CONTINUE
      ALPHA = DNEW/BOTTOM
      DO 80 I=1,N
      X(I) = X(I) + ALPHA*D(I)
   80 CONTINUE
```

```
C--------------------
C     FIND R(i+1)
C--------------------
      IF(ITER.EQ.50) THEN
      DO 100 I=1,N
      SUM = 0.
      DO 90 J=1,N
      SUM = SUM + A(I,J)*X(J)
   90 CONTINUE
      R(I) = B(I) - SUM
  100 CONTINUE
      ELSE
      DO 110 I=1,N
      R(I) = R(I) - ALPHA*Q(I)
  110 CONTINUE
      ENDIF

C--------------------
C     FIND S(i+1)
C--------------------
      DO 120 I=1,N
      S(I) = 0.
      S(I) = R(I)*P(I)
  120 CONTINUE

C--------------------
C     FIND BETA
C--------------------
      DOLD = DNEW

      DNEW = 0.
      DO 130 I=1,N
      DNEW = DNEW + R(I)*S(I)
  130 CONTINUE

      BETA = DNEW/DOLD

      DO 140 I=1,N
      D(I) = S(I) + BETA*D(I)
  140 CONTINUE

C-------------------------------------------------------
      ITER = ITER + 1
      GOTO 5

      ENDIF

      RETURN
      END

C-----------------------------------------------------------------C

C
C    SUBROUTINE FOR ASSEMBLING IN UPDATE
C
      SUBROUTINE ASSMUP(IE, INTMAT, RXELE, RYELE, SYSRX, SYSRY,
     *                  MXNEQ, MXELE  )

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION RXELE(3), RYELE(3)
      DIMENSION SYSRX(MXNEQ),SYSRY(MXNEQ)

      INTEGER INTMAT(MXELE,3)

C---------------------------------------------
C     ASSEMBLING SYSTEM LOAD VECTOR
C---------------------------------------------
      DO 200 I=1,3
      II = INTMAT(IE,I)
      SYSRX(II) = SYSRX(II) + RXELE(I)
      SYSRY(II) = SYSRY(II) + RYELE(I)
  200 CONTINUE

      RETURN
      END

C=================================================================
```

```
C
C     SUBROUTINE FOR ASSEMBLING IN PRESSURE EQUATIONS
C
      SUBROUTINE ASSMP(IE, INTMAT, AELE, RELE, SYSK, SYSR,
     *                 MXNEQ, MXELE )

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION AELE(3,3), RELE(3)
      DIMENSION SYSK(MXNEQ,MXNEQ),SYSR(MXNEQ)

      INTEGER INTMAT(MXELE,3)

C---------------------------------------------
C     ASSEMBLING SYSTEM STIFFNESS MATRIX
C---------------------------------------------

      DO 100 I=1,3
      DO 100 J=1,3
      II = INTMAT(IE,I)
      JJ = INTMAT(IE,J)
      SYSK(II,JJ) = SYSK(II,JJ) + AELE(I,J)
  100 CONTINUE

C---------------------------------------------
C     ASSEMBLING SYSTEM LOAD VECTOR
C---------------------------------------------

      DO 200 I=1,3
      II = INTMAT(IE,I)
      SYSR(II) = SYSR(II) + RELE(I)
  200 CONTINUE

      RETURN
      END

C=============================================================

C
C     SUBROUTINE FOR ASSEMBLING THE ELEMENT EQUATIONS
C
      SUBROUTINE ASSMBLE(IE, INTMAT, AELE, RXELE, RYELE, SYSK,
     *                   SYSRX, SYSRY, MXNEQ, MXELE)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION AELE(3,3), RXELE(3), RYELE(3)
      DIMENSION SYSK(MXNEQ,MXNEQ),SYSRX(MXNEQ), SYSRY(MXNEQ)

      INTEGER INTMAT(MXELE,3)

C---------------------------------------------
C     ASSEMBLING SYSTEM STIFFNESS MATRIX
C---------------------------------------------

      DO 100 I=1,3
      DO 100 J=1,3
      II = INTMAT(IE,I)
      JJ = INTMAT(IE,J)
      SYSK(II,JJ) = SYSK(II,JJ) + AELE(I,J)
  100 CONTINUE

C---------------------------------------------
C     ASSEMBLING SYSTEM LOAD VECTOR
C---------------------------------------------

      DO 200 I=1,3
      II = INTMAT(IE,I)
      SYSRX(II) = SYSRX(II) + RXELE(I)
      SYSRY(II) = SYSRY(II) + RYELE(I)
  200 CONTINUE

C---------------------------------------------

      RETURN
      END
```

ภาคผนวก ข
โปรแกรมคอมพิวเตอร์ STREAM

ข.1 ลักษณะของโปรแกรม STREAM

โปรแกรมคอมพิวเตอร์ STREAM ที่ได้ประดิษฐ์ขึ้น ดังที่จะได้แสดงรายละเอียดไว้ในหัวข้อ ข.2 นั้นประกอบด้วยโปรแกรมหลัก (main program) และ 7 โปรแกรมย่อย (subroutine) ลักษณะขั้นตอนที่สำคัญของโปรแกรมดังกล่าวประกอบไปด้วย

1. เริ่มต้นการทำงานโดยการอ่านข้อมูลของปัญหา ซึ่งประกอบไปด้วย จำนวนจุดต่อ, จำนวนเอลิเมนต์ของปัญหา, ตำแหน่งต่างๆของจุดต่อ และหมายเลขเอลิเมนต์พร้อมกับจุดต่อที่ประกอบเป็นเอลิเมนต์นั้นๆ ซึ่งจะอ่านค่าต่างๆเหล่านี้ในช่วงแรกของโปรแกรมหลัก [MAIN PROGRAM] พร้อมทั้งกำหนดค่าความเร็วที่แต่ละจุดต่อในโปรแกรมหลักนี้ด้วย

2. ทำการตรวจสอบว่ามีเอลิเมนต์ใดบ้างที่ไม่มีจุดดาวน์วินด์อยู่แล้วทำการแบ่งพื้นที่ของเอลิเมนต์ดังกล่าวออกเพื่อนำไปใช้ในการคำนวณเอลิเมนต์เมตริกซ์ต่อไป โดยเรียกโปรแกรมย่อย [SUBROUTINE CHK]

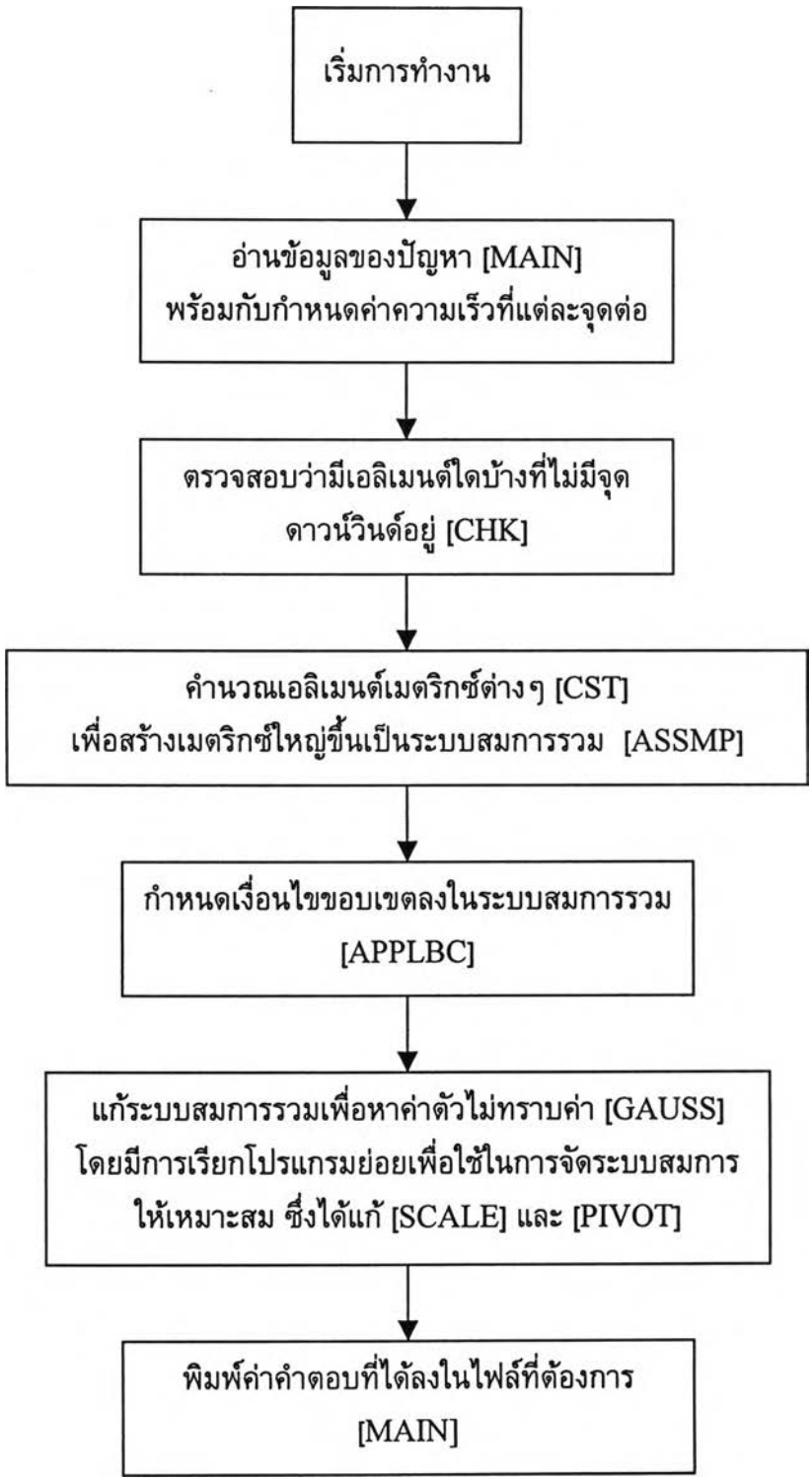
3. จากนั้นเรียกโปรแกรมย่อย [SUBROUTINE CST] เพื่อทำการคำนวณเอลิเมนต์เมตริกซ์ของปัญหาซึ่งมีทั้งเอลิเมนต์เมตริกซ์ของพจน์การแพร่และเอลิเมนต์เมตริกซ์ของพจน์การพา และส่งผ่านเอลิเมนต์เมตริกซ์เหล่านี้ไปสร้างเมตริกซ์ใหญ่ของระบบสมการรวมโดยเรียกโปรแกรมย่อย [SUBROUTINE ASSMP]

4. กำหนดเงื่อนไขขอบเขตลงในระบบสมการ เช่นมีบางจุดต่อที่ถูกกำหนดค่ามาไม่ให้เปลี่ยนแปลง โดยเรียกโปรแกรมย่อย [SUBROUTINE APPLYBC]

5. แก้ระบบสมการรวมเพื่อหาค่าตัวไม่ทราบค่าที่ทุกจุดต่อ โดยเรียกโปรแกรมย่อย [SUBROUTINE GAUSS] ซึ่งจะทำการแก้ระบบสมการด้วยวิธีกำจัดแบบเกาส์ โดยในโปรแกรมย่อยนี้จะมีการเรียกโปรแกรมย่อย [SUBROUTINE SCALE] เพื่อใช้สำหรับจัดสเกล และเรียกโปรแกรมย่อย [SUBROUTINE PIVOT] เพื่อใช้สำหรับเลือกตัวหลัก [10]

6. พิมพ์คำตอบของค่าที่คำนวณได้ลงในไฟล์ที่ต้องการ

โดยขั้นตอนต่างๆของโปรแกรมที่ได้อธิบายข้างต้นนี้สามารถสรุปดังแสดงในรูปที่ ข.1



รูปที่ ข.1 ลักษณะขั้นตอนการทำงานของโปรแกรม STREAM

ข.2 รายละเอียดของโปรแกรม STREAM

โปรแกรมคอมพิวเตอร์ STREAM ที่ประดิษฐ์ขึ้นในบทที่ 3 มีรายละเอียดดังนี้

```
C
C     PROGRAM STREAMLINE UPWIND FINITE ELEMENT
C

      PARAMETER (MXPOI=3500, MXELE=6500)
      PARAMETER (MXNEQ=MXPOI)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), TEXT(20), CHECKE(MXELE)
      DIMENSION U(MXPOI), V(MXPOI), P(MXPOI), CHECK(MXPOI)
      DIMENSION SYSK(MXNEQ,MXNEQ), SYSR(MXNEQ), SOL(MXNEQ)
      DIMENSION AREAVG(MXPOI)

      CHARACTER*20 NAME1, NAME2

      INTEGER  INTMAT(MXELE,3)
      INTEGER  IBCP(MXPOI)

      INTEGER(2) sthour, stminute, stsecond, sthund
      INTEGER(2) enhour, enminute, ensecond, enhund

C-------------------------------------------------
   10 WRITE(6,20)
   20 FORMAT(/, ' PLEASE ENTER INPUT FILE NAME:',/)
      READ(5, '(A)', ERR=10) NAME1

      OPEN(UNIT=7, FILE=NAME1, STATUS='OLD', ERR=10)
C-------------------------------------------------

      WRITE(6,25)
   25 FORMAT(/, ' Enter Peclet Number = ')
      READ(5,*) PE

C-------------------------------------------------

      CALL GETTIM(sthour, stminute, stsecond, sthund)

C----------------------------------
C     READ TEXT
C----------------------------------

      READ(7,*) NLINES
      DO 100 ILINE=1,NLINES
      READ(7,1) TEXT
    1 FORMAT(20A4)
  100 CONTINUE

C----------------------------------
C     READ INPUT DATA
C----------------------------------

      READ(7,1) TEXT
      READ(7,*) NPOI, NELEM
      IF(NPOI.GT.MXPOI) WRITE(6,110) NPOI
  110 FORMAT(/,' PLEASE INCREASE THE PARAMETER MXPOI TO',I5)
      IF(NPOI.GT.MXPOI) STOP
      IF(NELEM.GT.MXELE) WRITE(6,120) NELEM
  120 FORMAT(/,' PLEASE INCREASE THE PARAMETER MXELE TO',I5)
      IF(NELEM.GT.MXELE) STOP
```

```
C-------------------------------------------------------------
C     READ NODAL COORDINATES, BOUNDARY CONDITIONS, THEIR VALUES
C-------------------------------------------------------------

      READ(7,1) TEXT
      DO 150 IP = 1,NPOI
      READ(7,*)   I, IBCP(I), (COORD(I,K), K=1,2), P(I)
      IF(I.NE.IP) WRITE(6,155) IP
  155 FORMAT(/, ' NODE NO.', I5,' IN DATA FILE IS MISSING')
      IF(I.NE.IP) STOP
  150 CONTINUE

C--------------------------------------
C     READ ELEMENT NODAL CONNECTION
C--------------------------------------

      READ(7,1) TEXT
      DO 160 IE=1,NELEM
      READ(7,*)   I, (INTMAT(I,J), J=1,3)
      IF(I.NE.IE) WRITE(6,165) IE
  165 FORMAT(/, ' ELEMENT NO.', I5, ' IN DATA FILE IS MISSING')
      IF(I.NE.IE) STOP
  160 CONTINUE

C-----------------------------
C     PRINT OUT TITLE
C-----------------------------

      WRITE(6,200) NPOI, NELEM
  200 FORMAT(/,'   THE FINITE ELEMENT MODEL CONSISTS OF :',/,
     *         '      NUMBER OF NODES              =', I6, /,
     *         '      NUMBER OF ELEMENTS           =', I6, /)

C--------------------------------------
C     RESET THE SYSTEM OF EQUATIONS
C--------------------------------------

      NEQ = NPOI
      DO 520 I=1,NEQ
        SYSR(I)=0.
      DO 520 J=1,NEQ
        SYSK(I,J)=0.
  520 CONTINUE

C--------------------------------------
C     DEFINE THE VALUE OF VELOCITIES
C--------------------------------------
      DO 30 I =1,NPOI

        X = COORD(I,1)
        Y = COORD(I,2)

        AREAVG(I) = 0.

        U(I) =  2.*Y*(1.-(X*X))
        V(I) = -2.*X*(1.-(Y*Y))

   30 CONTINUE

C-------------------------------------------
C     CHECK EVERY ELEMENT FOR ADD AREA
C-------------------------------------------

      CALL CHK(NELEM,  COORD, INTMAT,  MXPOI,
     *         MXELE, AREAVG,      U,      V)
```

```
C-------------------------------------------
C     CALCULATION FOR PSI COEFFICIENT
C-------------------------------------------

      CALL CST( NELEM, COORD, INTMAT, PE, SYSK,   SYSR,
     *          MXPOI, MXELE,  MXNEQ,  U,    V, AREAVG)

C-----------------------------------------------
C      APPLY BOUNDARY CONDITIONS FOR VELOCITY
C-----------------------------------------------

      CALL APPLYBC(NPOI, IBCP,      P, SYSK,
     *             SYSR, MXPOI, MXNEQ )

C-----------------------------------------------
C     SOLVE A SET OF SIMULTANEOUS EQUATIONS
C-----------------------------------------------

      CALL GAUSS(NEQ, SYSK, SYSR, SOL, MXNEQ)

      DO 330 I=1,NPOI
        IF(ABS(SOL(I)).LE.1.E-10) SOL(I) = 0.
        P(I) = SOL(I)
  330 CONTINUE

      CALL GETTIM(enhour, enminute, ensecond, enhund)

c-------------------------------------------------------------
      WRITE(6,300) sthour, stminute, stsecond, sthund
  300 FORMAT('  Start time :',3X,I2,':',I2.2,':',I2.2,':',I2.2)
      WRITE(6,310) enhour, enminute, ensecond, enhund
  310 FORMAT('    End time :',3X,I2,':',I2.2,':',I2.2,':',I2.2)
c-------------------------------------------------------------

C---------------------------------
C     PRINT OUT SOLUTION
C---------------------------------

  930 WRITE(6,940)
  940 FORMAT(/, ' ENTER THE OUTPUT FILE NAME',/)
      READ(5 , '(A)', ERR=930 ) NAME2
      OPEN(UNIT=8, FILE=NAME2 , STATUS='NEW',ERR=930)

      DO 1000 I=1,NPOI
        WRITE(8,950) I, P(I)
  950   FORMAT(I6,2X,E12.4)
 1000 CONTINUE

C-------------------------------------------------------------

      STOP
      END

C
C-------------------------------------------------------------
C

      SUBROUTINE CHK(NELEM, COORD, INTMAT,  MXPOI, IBCU,
     *               IBCV, MXELE, AREAVG,      U,    V)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2)
      DIMENSION U(MXPOI), V(MXPOI), AREAVG(MXPOI)

      INTEGER   INTMAT(MXELE,3)

C-------------------------------------------------------------
```

```
C
C     LOOP OVER THE NUMBER OF ELEMENTS :
C

      DO 500 IE=1,NELEM

      CHECK3 = 0.

C
C     FIND ELEMENT LOCAL COORDINATES :
C
      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)

      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))

      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *          ' HAS NEGATIVE OR ZERO AREA ', /,
     *          ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *          ' AND ELEMENT NODAL CONNECTIONS ---'          )
      IF(AREA.LE.0.) STOP

C-------------------------
C     CHECK FOR NODE 1
C-------------------------
      UJ   =  U(II)
      VJ   =  V(II)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))
        CHECK2 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))
          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF
      ENDIF
C-------------------------
C     CHECK FOR NODE 2
C-------------------------
      UJ   =  U(JJ)
      VJ   =  V(JJ)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))
        CHECK2 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))
          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF
      ENDIF
C-------------------------
C     CHECK FOR NODE 3
C-------------------------
      UJ   =  U(KK)
      VJ   =  V(KK)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN
        CHECK1 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))
        CHECK2 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))
          IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN
            CHECK3 = 1.
          ENDIF
      ENDIF
```

```
      IF(CHECK3.NE.1.) THEN
         DIVIDE = 3.
         DO 10 I=1,3
         NN = INTMAT(IE,I)
           IF(U(NN).EQ.0..AND.V(NN).EQ.0.) THEN
              DIVIDE = DIVIDE - 1.
           ENDIF
   10    CONTINUE
           IF(DIVIDE.NE.0.) THEN
              DO 15 I=1,3
              NN = INTMAT(IE,I)
                 IF(U(NN).EQ.0..AND.V(NN).EQ.0.) THEN
                    AREAVG(NN) = 0.
                 ELSE
                    AREAVG(NN) = AREAVG(NN) + (AREA/DIVIDE)
                 ENDIF
   15         CONTINUE
           ENDIF
      ENDIF

C-----------------------------------------------------------------

  500 CONTINUE

      RETURN
      END
C
C-----------------------------------------------------------------
C

      SUBROUTINE CST( NELEM, COORD, INTMAT, PE, SYSK,   SYSR,
     *                MXPOI, MXELE,  MXNEQ,  U,    V, AREAVG)

      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION COORD(MXPOI,2), SYSK(MXNEQ,MXNEQ)
      DIMENSION SYSR(MXNEQ)
      DIMENSION U(MXPOI), V(MXPOI), AREAVG(MXPOI)
      DIMENSION A(3,3), R(3)
      DIMENSION ACOV(3,3), ADIFX(3,3), ADIFY(3,3)
      INTEGER INTMAT(MXELE,3)
C
C     LOOP OVER THE NUMBER OF ELEMENTS :
C
      DO 500 IE=1,NELEM

C
C     FIND ELEMENT LOCAL COORDINATES :
C
      II = INTMAT(IE,1)
      JJ = INTMAT(IE,2)
      KK = INTMAT(IE,3)

      XG1 = COORD(II,1)
      XG2 = COORD(JJ,1)
      XG3 = COORD(KK,1)
      YG1 = COORD(II,2)
      YG2 = COORD(JJ,2)
      YG3 = COORD(KK,2)

      AREA = 0.5*(XG2*(YG3-YG1)+XG1*(YG2-YG3)+XG3*(YG1-YG2))
      IF(AREA.LE.0.) WRITE(6,5) IE
    5 FORMAT(/,'   !!! ERROR !!! ELEMENT NO.', I5,
     *         ' HAS NEGATIVE OR ZERO AREA ', /,
     *         ' --- CHECK F.E. MODEL FOR NODAL COORDINATES',
     *         ' AND ELEMENT NODAL CONNECTIONS ---'         )
      IF(AREA.LE.0.) STOP
```

```
      B1 = YG2 - YG3
      B2 = YG3 - YG1
      B3 = YG1 - YG2
      C1 = XG3 - XG2
      C2 = XG1 - XG3
      C3 = XG2 - XG1

C=================================
C     SET UP [ADIF] METRIX :
C=================================
      DO 30 I = 1,3
      DO 30 J = 1,3
        A(I,J)    = 0.
        ADIFX(I,J) = 0.
        ADIFY(I,J) = 0.
        ACOV(I,J)  = 0.
   30 CONTINUE

C=================================
C     SET UP [ADIFX] METRIX :
C=================================
      ADIFX(1,1) = (B1*B1)/(4.*AREA)
      ADIFX(1,2) = (B1*B2)/(4.*AREA)
      ADIFX(1,3) = (B1*B3)/(4.*AREA)

      ADIFX(2,1) = ADIFX(1,2)
      ADIFX(2,2) = (B2*B2)/(4.*AREA)
      ADIFX(2,3) = (B2*B3)/(4.*AREA)

      ADIFX(3,1) = ADIFX(1,3)
      ADIFX(3,2) = ADIFX(2,3)
      ADIFX(3,3) = (B3*B3)/(4.*AREA)

C=================================
C     SET UP [ADIFY] METRIX :
C=================================
      ADIFY(1,1) = (C1*C1)/(4.*AREA)
      ADIFY(1,2) = (C1*C2)/(4.*AREA)
      ADIFY(1,3) = (C1*C3)/(4.*AREA)

      ADIFY(2,1) = ADIFY(1,2)
      ADIFY(2,2) = (C2*C2)/(4.*AREA)
      ADIFY(2,3) = (C2*C3)/(4.*AREA)

      ADIFY(3,1) = ADIFY(1,3)
      ADIFY(3,2) = ADIFY(2,3)
      ADIFY(3,3) = (C3*C3)/(4.*AREA)

C=================================
C     SET UP [A] METRIX :
C=================================

C=============================================================
C     FIND MASS FLOW RATE PAST THE SIDE OF ELEMENT
C       F1 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 1
C       F2 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 2
C       F3 = MASS FLOW ON THE SIDE THAT OPPOSITE NODE 3
C=============================================================
      F1 = -(XG3-XG2)*(0.5*(V(JJ)+V(KK)))+(YG3-YG2)*(0.5*(U(JJ)+U(KK)))
      F2 = -(XG1-XG3)*(0.5*(V(KK)+V(II)))+(YG1-YG3)*(0.5*(U(KK)+U(II)))
      F3 = -(XG2-XG1)*(0.5*(V(II)+V(JJ)))+(YG2-YG1)*(0.5*(U(II)+U(JJ)))

C=====================================
C     FIND STREAMLINE UPWIND NODE
C     SET UP [ACOV] METRIX :
C=====================================

      COEF = 0.
```

```
C-------------------------
C     CHECK FOR NODE 1
C-------------------------

      UJ   =  U(II)
      VJ   =  V(II)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))
      CHECK2 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))

         IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

            IF(F1.LT.0.) THEN

               IF(F2.LT.0..AND.F3.GT.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F3.LT.0..AND.F2.GT.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F2.GT.0..AND.F3.EQ.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F3.GT.0..AND.F2.EQ.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F2.EQ.F3) THEN
                  FP = 0.5
                  FN = 0.5
               ELSEIF(ABS(F2).GE.ABS(F1)) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(ABS(F3).GE.ABS(F1)) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F2.GT.0..AND.F3.GT.0.) THEN

                  IF(U(JJ).EQ.0..AND.V(JJ).EQ.0.) THEN
                     FP = 0.
                     FN = 1.
                  ELSEIF(U(KK).EQ.0..AND.V(KK).EQ.0.) THEN
                     FP = 1.
                     FN = 0.
                  ELSE
                     FP = ABS(F2/F1)
                     FN = 1.-FP
                  ENDIF

               ENDIF

               X = FP*XG2 + FN*XG3
               Y = FP*YG2 + FN*YG3

               US   = SQRT((UJ*UJ)+(VJ*VJ))
               DS   = SQRT((X-XG1)*(X-XG1)+(Y-YG1)*(Y-YG1))
               AREA = AREA + AREAVG(II)
               COEF = (US*AREA)/DS

               ACOV(1,1) =     COEF
               ACOV(1,2) = -FP*COEF
               ACOV(1,3) = -FN*COEF

            ENDIF
         ENDIF
      ENDIF
```

```
C-------------------------
C     CHECK FOR NODE 2
C-------------------------

      UJ   =  U(JJ)
      VJ   =  V(JJ)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG2-XG1))+(UJ*(YG2-YG1))
      CHECK2 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))

         IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

            IF(F2.LT.0.) THEN

               IF(F3.LT.0..AND.F1.GT.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F1.LT.0..AND.F3.GT.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F3.GT.0..AND.F1.EQ.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F1.GT.0..AND.F3.EQ.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F1.EQ.F3) THEN
                  FP = 0.5
                  FN = 0.5
               ELSEIF(ABS(F1).GE.ABS(F2)) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(ABS(F3).GE.ABS(F2)) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F1.GT.0..AND.F3.GT.0.) THEN

                  IF(U(II).EQ.0..AND.V(II).EQ.0.) THEN
                     FP = 1.
                     FN = 0.
                  ELSEIF(U(KK).EQ.0..AND.V(KK).EQ.0.) THEN
                     FP = 0.
                     FN = 1.
                  ELSE
                     FP = ABS(F3/F2)
                     FN = 1.-FP
                  ENDIF

               ENDIF

               X = FP*XG3 + FN*XG1
               Y = FP*YG3 + FN*YG1

               US   = SQRT((UJ*UJ)+(VJ*VJ))
               DS   = SQRT((X-XG2)*(X-XG2)+(Y-YG2)*(Y-YG2))
               AREA = AREA + AREAVG(JJ)
               COEF = (US*AREA)/DS

               ACOV(2,1) = -FN*COEF
               ACOV(2,2) =     COEF
               ACOV(2,3) = -FP*COEF

            ENDIF
         ENDIF
      ENDIF
```

```
C-------------------------
C     CHECK FOR NODE 3
C-------------------------

      UJ    =  U(KK)
      VJ    =  V(KK)
      IF(UJ.NE.0..OR.VJ.NE.0.) THEN

      CHECK1 = -(VJ*(XG3-XG2))+(UJ*(YG3-YG2))
      CHECK2 = -(VJ*(XG1-XG3))+(UJ*(YG1-YG3))

         IF(CHECK1.GE.0..AND.CHECK2.GE.0.) THEN

            IF(F3.LT.0.) THEN

               IF(F1.LT.0..AND.F2.GT.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F2.LT.0..AND.F1.GT.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F1.GT.0..AND.F2.EQ.0.) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(F2.GT.0..AND.F1.EQ.0.) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F1.EQ.F2) THEN
                  FP = 0.5
                  FN = 0.5
               ELSEIF(ABS(F1).GE.ABS(F3)) THEN
                  FP = 1.
                  FN = 0.
               ELSEIF(ABS(F2).GE.ABS(F3)) THEN
                  FP = 0.
                  FN = 1.
               ELSEIF(F1.GT.0..AND.F2.GT.0.) THEN

                  IF(U(II).EQ.0..AND.V(II).EQ.0.) THEN
                     FP = 0.
                     FN = 1.
                  ELSEIF(U(JJ).EQ.0..AND.V(JJ).EQ.0.) THEN
                     FP = 1.
                     FN = 0.
                  ELSE
                     FP = ABS(F1/F3)
                     FN = 1.-FP
                  ENDIF

               ENDIF

               X = FN*XG2 + FP*XG1
               Y = FN*YG2 + FP*YG1

               US   = SQRT((UJ*UJ)+(VJ*VJ))
               DS   = SQRT((X-XG3)*(X-XG3)+(Y-YG3)*(Y-YG3))
               AREA = AREA + AREAVG(KK)
               COEF = (US*AREA)/DS

               ACOV(3,1) = -FP*COEF
               ACOV(3,2) = -FN*COEF
               ACOV(3,3) =     COEF

            ENDIF
         ENDIF
      ENDIF
```

```
C-------------------------------------
C     SET THE VALUE FOR
C     THE COEFFICIENT OF DIFFUSION TERM
C-------------------------------------
      COEF1 = 1./(PE)
      COEF2 = 1./(PE)

C=====================================================
C     SET UP [A] METRIX :
C     [A] = [ACOV] + COEF1[ADIFX] + COEF2[ADIFY]
C=====================================================
      DO 50 I = 1,3
      DO 50 J = 1,3
      A(I,J) = ACOV(I,J) + COEF1*ADIFX(I,J) + COEF2*ADIFY(I,J)
   50 CONTINUE

C===================================
C     SET UP [R] METRIX :
C===================================
      DO 160 I=1,3
      R(I) = 0.
  160 CONTINUE

      CALL ASSMP(  IE, INTMAT,     A,     R,
     *           SYSK,   SYSR, MXELE, MXNEQ)


  500 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------

      SUBROUTINE APPLYBC(NPOI,  IBCP,     P, SYSK,
     *                   SYSR, MXPOI, MXNEQ)
C
C     APPLY BOUNDARY CONDITIONS BEFORW SOLVING FOR NODAL
C     WITH CONDITION CODES OF:
C        0 = FREE TO CHANGE
C        1 = FIXED AS SPECIFIED
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION SYSK(MXNEQ,MXNEQ), SYSR(MXNEQ)
      DIMENSION P(MXPOI)
      INTEGER IBCP(MXPOI)
C
C     APPLY BOUNDARY CONDITIONS FOR NODAL U-VELOCITIES :
C
      DO 100 IEQ=1,NPOI
        IF(IBCP(IEQ).EQ.0) GOTO 100
           DO 110 IR=1,NPOI
             IF(IR.EQ.IEQ) GOTO 110
             SYSR(IR) = SYSR(IR) - SYSK(IR,IEQ)*P(IEQ)
             SYSK(IR,IEQ) = 0.
  110      CONTINUE
           DO 120 IC=1,NPOI
             SYSK(IEQ,IC) = 0.
  120      CONTINUE
        SYSK(IEQ,IEQ) = 1.
        SYSR(IEQ) = P(IEQ)
  100 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------
```

```
      SUBROUTINE ASSMP(IE, INTMAT, CK, R, SYSK, SYSR,
     *                 MXELE, MXNEQ   )

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION CK(3,3), R(3)
      DIMENSION SYSK(MXNEQ,MXNEQ),SYSR(MXNEQ)
      INTEGER INTMAT(MXELE,3)
C
C     ASSEMBLING SYSTEM STIFFNESS MATRIX
C
C     CONTRIBUTION OF COEFFICIENTS ASSOCIATED WITH PRESSURE:
C
      DO 100 I=1,3
      DO 100 J=1,3
        II = INTMAT(IE,I)
        JJ = INTMAT(IE,J)
        SYSK(II,JJ) = SYSK(II,JJ) + CK(I,J)
  100 CONTINUE

C
C     ASSEMBLING SYSTEM LOAD VECTOR
C
C     CONTRIBUTION OF COEFFICIENTS ASSOCIATED WITH PRESSURE:
C
      DO 200 I=1,3
        II = INTMAT(IE,I)
        SYSR(II) = SYSR(II) + R(I)
  200 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------

      SUBROUTINE GAUSS(N, A, B, X, MXNEQ)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ), X(MXNEQ)

      CALL SCALE(N, A, B, MXNEQ)
      DO 100 IP=1,N-1
        CALL PIVOT(N, A, B, MXNEQ, IP)
          DO 200 IE=IP+1,N
            RATIO = A(IE,IP)/A(IP,IP)
              DO 300 IC=IP+1,N
                A(IE,IC) = A(IE,IC) - RATIO*A(IP,IC)
  300         CONTINUE
              B(IE) = B(IE) - RATIO*B(IP)
  200     CONTINUE
          DO 400 IE=IP+1,N
            A(IE,IP) = 0.
  400     CONTINUE
  100 CONTINUE
      X(N) = B(N)/A(N,N)
      DO 500 IE=N-1,1,-1
        SUM = 0.
          DO 600 IC=IE+1,N
            SUM = SUM + A(IE,IC)*X(IC)
  600     CONTINUE
        X(IE) = (B(IE) - SUM)/A(IE,IE)
  500 CONTINUE

      RETURN
      END

C-----------------------------------------------------------
```

```
      SUBROUTINE SCALE(N, A, B, MXNEQ)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ)
      DO 10 IE=1,N
        BIG = ABS(A(IE,1))
          DO 20 IC=2,N
            AMAX = ABS(A(IE,IC))
            IF(AMAX.GT.BIG) BIG = AMAX
   20     CONTINUE
          DO 30 IC=1,N
            A(IE,IC) = A(IE,IC)/BIG
   30     CONTINUE
        B(IE) = B(IE)/BIG
   10 CONTINUE

      RETURN
      END
C-------------------------------------------------------

      SUBROUTINE PIVOT(N, A, B, MXNEQ, IP)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ)
      JP = IP
      BIG = ABS(A(IP,IP))
      DO 10 I=IP+1,N
        AMAX = ABS(A(I,IP))
          IF(AMAX.GT.BIG) THEN
            BIG = AMAX
            JP  = I
          ENDIF
   10 CONTINUE
      IF(JP.NE.IP) THEN
        DO 20  J=IP,N
          DUMY    = A(JP,J)
          A(JP,J) = A(IP,J)
          A(IP,J) = DUMY
   20   CONTINUE
        DUMY    = B(JP)
        B(JP)   = B(IP)
        B(IP)   = DUMY
      ENDIF

      RETURN
      END
C-------------------------------------------------------------
```

ข.3 ลักษณะของข้อมูลที่โปรแกรม STREAM ต้องการ

ลักษณะข้อมูลที่โปรแกรมไฟไนต์เอลิเมนต์สตรีมไลน์อัปวินด์ต้องการนี้ สามารถจำแนกออกเป็น 6 ส่วนย่อยได้ดังนี้

ส่วนที่ 1 ประโยคอธิบายกำกับไฟล์ข้อมูล

บรรทัดแรก	ตัวเลขระบุจำนวนบรรทัดที่เป็นตัวอักษร
บรรทัดต่อไป	ประโยคต่างๆ ที่มีจำนวนบรรทัดเท่าที่ระบุไว้

ตัวอย่างเช่น :

```
2
ADVECTION SKEW TO THE MESH PROBLEM
TEST FOR PURE CONVECTION
```

ส่วนที่ 2 ขนาดของปัญหา

บรรทัดแรก คำระบุจำนวนจุดต่อ และจำนวนเอลิเมนต์

บรรทัดที่ 2 ตัวเลขจำนวนจุดต่อ และจำนวนเอลิเมนต์

ตัวอย่างเช่น :

NPOIN	NELEM
496	900

ส่วนที่ 3 ลักษณะของจุดต่อ

บรรทัดแรก คำระบุลักษณะของจุดต่อ

บรรทัดที่ต่อๆไป ตัวเลขแสดงหมายเลขจุดต่อ เงื่อนไขขอบเขตของตัวไม่ทราบค่า และตำแหน่งจุดต่อในแกน x และ y และค่าของตัวไม่ทราบค่าที่จุดต่อ

ตัวอย่างเช่น :

NODE	IBCP	COORD-X	COORD-Y	P
1	1	-1.	0.	4.1224E-09
2	1	-0.93333	0.	5.9336E-08
⋮	⋮	⋮	⋮	⋮
494	1	0.86667	1.	4.1224E-09
495	1	0.93333	1.	4.1224E-09
496	1	1.	1.	4.1224E-09

หมายเหตุ: เงื่อนไขของเขต IBCP ของจุดต่อหมายถึง

IBCP = 1 จุดต่อนั้นถูกกำหนดไม่ให้ค่าที่กำหนดไว้เปลี่ยนแปลง

IBCP = 0 ให้ทำการคำนวณหาค่าที่จุดต่อเหล่านี้

ส่วนที่ 4 ลักษณะของเอลิเมนต์

บรรทัดแรก คำระบุลักษณะของเอลิเมนต์

บรรทัดที่ต่อๆไป หมายเลขจุดต่อทั้งสามในทิศทางทวนเข็มนาฬิกาที่ประกอบขึ้นเป็นเอลิเมนต์นั้นๆ

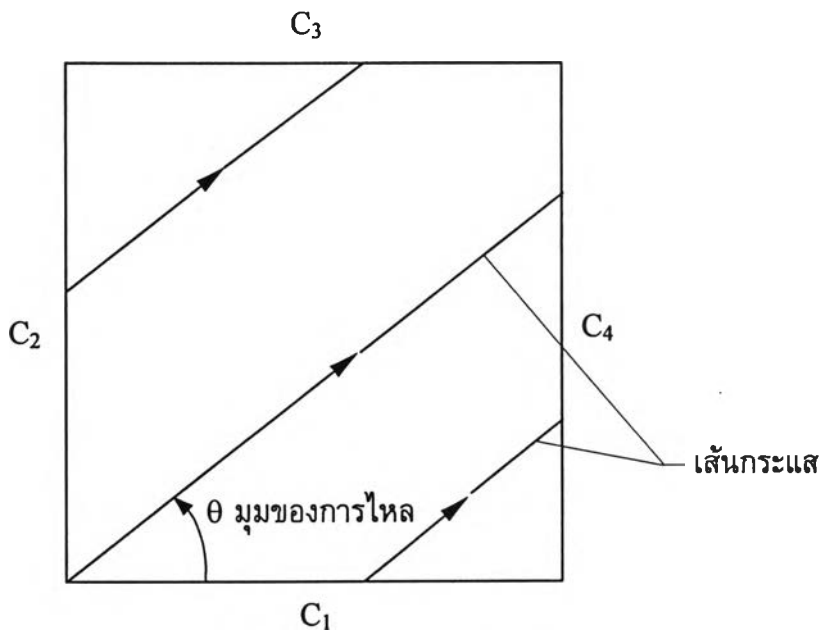
ตัวอย่างเช่น :

ELEMENT NO.	I	J	K
1	1	2	33
2	1	33	32
⋮	⋮	⋮	⋮
898	495	464	465
899	495	494	464
900	496	495	465

เพื่อให้เกิดความเข้าใจกับลักษณะของข้อมูลที่โปรแกรม STREAM ต้องการได้ดียิ่งขึ้น จะได้แสดงตัวอย่างการใช้โปรแกรมในหัวข้อต่อไปนี้

ข.4 ตัวอย่างการใช้โปรแกรม STREAM

ในหัวข้อนี้จะได้ยกตัวอย่างปัญหาที่ 1 ในบทที่ 3 ซึ่งเป็นปัญหาที่มีเฉพาะพจน์การพาเพียงอย่างเดียว โดยที่ขอบเขตของปัญหามีลักษณะเป็นสี่เหลี่ยมจัตุรัสดังแสดงในรูปที่ ข.2 และกำหนดให้ความเร็วมีลักษณะคงตัว (uniform) ตลอดภายในขอบเขตของปัญหา โดยที่มีมุมการไหลที่แตกต่างกัน แต่ในที่นี้จะแสดงเฉพาะการไหลที่ทำมุม 45 องศาเท่านั้น และรูปแบบจำลองไฟไนต์เอลิเมนต์ของปัญหานี้ได้แสดงไว้ในรูปที่ ข.3 ซึ่งประกอบไปด้วย 36 จุดต่อ และ 50 เอลิเมนต์เท่านั้นเพื่อให้สามารถ



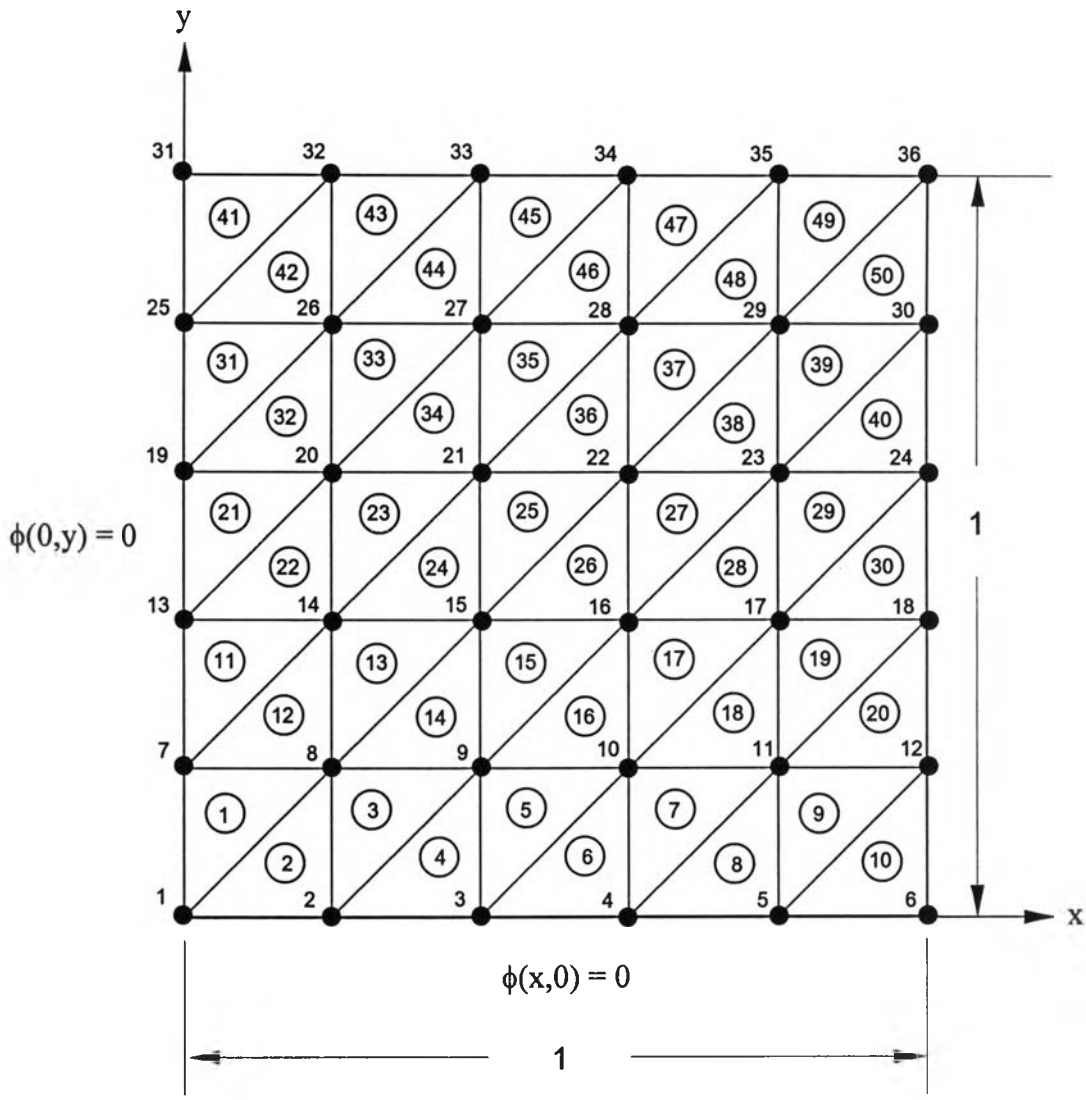
รูปที่ ข.2 ขอบเขตของปัญหาตัวอย่าง

สมการเชิงอนุพันธ์สำหรับปัญหาตัวอย่าง

$$u\frac{\partial \phi}{\partial x} + v\frac{\partial \phi}{\partial y} = 0$$

เงื่อนไขขอบเขตของปัญหา

กำหนดให้ค่า ϕ ตลอดขอบ C_1 มีค่าเท่ากับศูนย์ และตลอดขอบ C_2 มีค่าเท่ากับหนึ่ง สำหรับด้านขอบ C_3 และ C_4 นั้นไม่ต้องทำการกำหนดค่า



รูปที่ ข.3 รูปแบบจำลองไฟไนต์เอลิเมนต์และเงื่อนไขขอบเขต

ข้อมูลที่สอดคล้องกับรูปลักษณะไฟไนต์เอลิเมนต์ในรูปที่ ข.3 โดยสมมติว่าอยู่ในไฟล์ชื่อ 'EX1.DAT' ซึ่งมีรายละเอียดดังต่อไปนี้

```
2
FINITE ELEMENT DATA
FOR ADVECTION SKEW TO THE MESH PROBLEM
   NPOIN     NELEM
      36        50
  NODE     IBCP    COORD-X   COORD-Y    P
    1        1        0.        0.       1.
    2        1       0.2        0.       0.
    3        1       0.4        0.       0.
    4        1       0.6        0.       0.
    5        1       0.8        0.       0.
    6        1        1.        0.       0.
    7        1        0.       0.2       1.
    8        0       0.2       0.2       0.
    9        0       0.4       0.2       0.
   10        0       0.6       0.2       0.
   11        0       0.8       0.2       0.
   12        0        1.       0.2       0.
```

13	1	0.	0.4	1.
14	0	0.2	0.4	0.
15	0	0.4	0.4	0.
16	0	0.6	0.4	0.
17	0	0.8	0.4	0.
18	0	1.	0.4	0.
19	1	0.	0.6	1.
20	0	0.2	0.6	0.
21	0	0.4	0.6	0.
22	0	0.6	0.6	0.
23	0	0.8	0.6	0.
24	0	1.	0.6	0.
25	1	0.	0.8	1.
26	0	0.2	0.8	0.
27	0	0.4	0.8	0.
28	0	0.6	0.8	0.
29	0	0.8	0.8	0.
30	0	1.	0.8	0.
31	1	0.	1.	1.
32	0	0.2	1.	0.
33	0	0.4	1.	0.
34	0	0.6	1.	0.
35	0	0.8	1.	0.
36	0	1.	1.	0.

ELEMENT NO.	I	J	K
1	1	8	7
2	1	2	8
3	2	9	8
4	2	3	9
5	3	10	9
6	3	4	10
7	4	11	10
8	4	5	11
9	5	12	11
10	5	6	12
11	7	14	13
12	7	8	14
13	8	15	14
14	8	9	15
15	9	16	15
16	9	10	16
17	10	17	16
18	10	11	17
19	11	18	17
20	11	12	18
21	13	20	19
22	13	14	20
23	14	21	20
24	14	15	21
25	15	22	21
26	15	16	22
27	16	23	22
28	16	17	23
29	17	24	23
30	17	18	24
31	19	26	25
32	19	20	26
33	20	27	26
34	20	21	27
35	21	28	27
36	21	22	28

```
37          22          29          28
38          22          23          29
39          23          30          29
40          23          24          30
41          25          32          31
42          25          26          32
43          26          33          32
44          26          27          33
45          27          34          33
46          27          28          34
47          28          35          34
48          28          29          35
49          29          36          35
50          29          30          36
```

รูปที่ ข.4 ข้อมูลในไฟล์ชื่อ 'EX1.DAT'

เมื่อผู้ใช้ทำการคำนวณโดยใช้โปรแกรม STREAM โปรแกรมจะถามชื่อไฟล์ข้อมูลซึ่งผู้ใช้จะพิมพ์ตอบกลับไป จากนั้นจะให้ป้อนค่าของตัวแปรไร้มิติหรือคือค่าเพกเลตนัมเบอร์ (Peclet number) ซึ่งในที่นี้เป็นปัญหาที่มีพจน์การพาเพียงอย่างเดียว ดังนั้นจะต้องใส่ค่าของเพกเลตนัมเบอร์ให้มีค่าเข้าใกล้อนันต์ จากนั้นโปรแกรมจะทำการคำนวณตามขั้นตอนที่ได้อธิบายไปข้างต้น เมื่อการคำนวณสิ้นสุดลง โปรแกรมจะให้ผู้ใช้ใส่ชื่อไฟล์ที่จะบรรจุผลลัพธ์ โดยที่ขั้นตอนดังกล่าวจะปรากฏบนจอคอมพิวเตอร์ดังแสดงในรูปที่ ข.5

```
>STREAM     <Enter>

PLEASE ENTER INPUT FILE NAME:
EX1.DAT

Enter Peclet Number =
1.E+59

  THE FINITE ELEMENT MODEL CONSISTS OF :
     NUMBER OF NODES                =    36
     NUMBER OF ELEMENTS             =    50

 Start time :   16:07:05:54
   End time :   16:07:06:09

ENTER THE OUTPUT FILE NAME
EX1.OUT

Stop - Program terminated.
```

รูปที่ ข.5 ลำดับขั้นตอนที่ปรากฏบนจอคอมพิวเตอร์ในขณะใช้โปรแกรม STREAM

ผลลัพธ์ที่ได้จากการคำนวณซึ่งบรรจุอยู่ในไฟล์ชื่อ 'EX1.OUT' ได้แสดงในรูปที่ ข.6

```
          1     .1000E+01
          2     .0000E+00
          3     .0000E+00
          4     .0000E+00
          5     .0000E+00
```

```
 6      .0000E+00
 7      .1000E+01
 8      .1000E+01
 9      .0000E+00
10      .0000E+00
11      .0000E+00
12      .0000E+00
13      .1000E+01
14      .1000E+01
15      .1000E+01
16      .0000E+00
17      .0000E+00
18      .0000E+00
19      .1000E+01
20      .1000E+01
21      .1000E+01
22      .1000E+01
23      .0000E+00
24      .0000E+00
25      .1000E+01
26      .1000E+01
27      .1000E+01
28      .1000E+01
29      .1000E+01
30      .0000E+00
31      .1000E+01
32      .1000E+01
33      .1000E+01
34      .1000E+01
35      .1000E+01
36      .1000E+01
```

รูปที่ ข.6 ลักษณะของไฟล์ผลลัพธ์ที่อยู่ในไฟล์ 'EX1.OUT'

ภาคผนวก ค
โปรแกรมคอมพิวเตอร์ PCG

ค.1 ลักษณะของโปรแกรม PCG

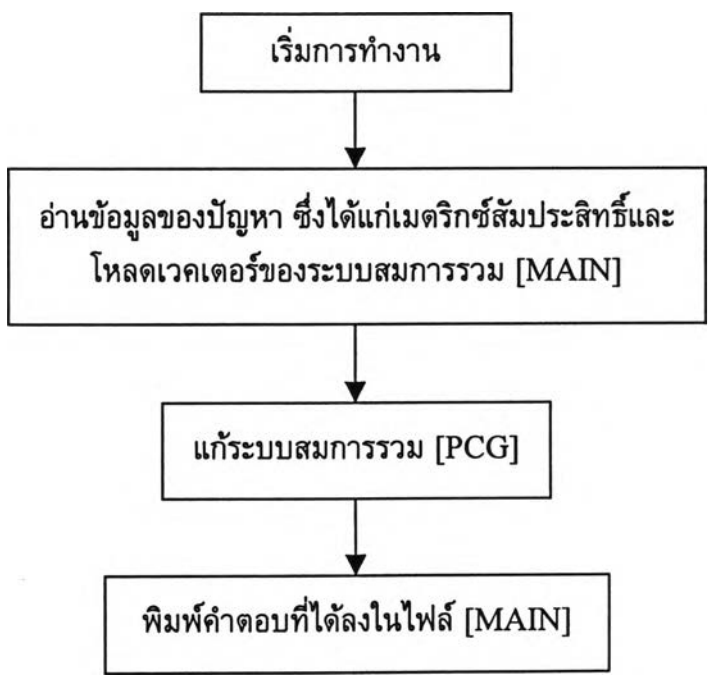
โปรแกรมคอมพิวเตอร์ PCG ที่ได้ประดิษฐ์ขึ้นนี้ ประกอบไปด้วยโปรแกรมหลัก (main program) และ 1 โปรแกรมย่อย (subroutine) ซึ่งได้แสดงรายละเอียดไว้ในหัวข้อ ค.2 และลักษณะของโปรแกรมดังกล่าวประกอบไปด้วยขั้นตอนดังต่อไปนี้

1. เริ่มต้นการทำงานโดยการอ่านข้อมูลของเมตริกซ์สัมประสิทธิ์พร้อมทั้งโหลดเวคเตอร์ของระบบสมการที่จะทำการแก้ อีกทั้งยังให้ผู้ใช้ทำการใส่ค่าความผิดพลาดที่ยอมให้ได้ และจำนวนรอบสูงสุดในการคำนวณแบบวนซ้ำ ซึ่งจะทำการอ่านในช่วงแรกของโปรแกรมหลัก [MAIN PROGRAM]

2. จากนั้นจะทำการแก้ระบบสมการด้วยระเบียบวิธีคอนจูเกตเกรเดียนท์โดยเรียกโปรแกรมย่อย [SUBROUTINE PCG]

3. พิมพ์คำตอบของระบบสมการลงในไฟล์ที่ต้องการ

ลำดับขั้นตอนดังกล่าวสามารถสรุปดังแสดงในรูปที่ ค.1



รูปที่ ค.1 ลักษณะขั้นตอนการทำงานของโปรแกรม PCG

ค.2 รายละเอียดของโปรแกรม PCG

โปรแกรมคอมพิวเตอร์ PCG ที่ประดิษฐ์ขึ้นในบทที่ 5 มีรายละเอียดดังนี้

```
C
C      This program solves a set of N simulations equations
C      using the Preconditioning Conjugate Gradient Method
C
C      The equation is in the form  Ax = b
C      where x is an unknown vector
C            b is a known vector
C            A is a known, square, sysmmetric and Positive-definite
C              or Positive-indefinite matrix
C

      PARAMETER (MXNEQ=500)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ), X(MXNEQ)
      CHARACTER*20 NAME1, NAME2
      INTEGER(2)  tmphour, tmpminute, tmpsecond, tmphund

C-----------------------------------------------------------------------

   10 WRITE(6,23)
   23 FORMAT(/, ' PLEASE ENTER INPUT FILE NAME:',/)
      READ(5, '(A)', ERR=10) NAME1
      OPEN(UNIT=7, FILE=NAME1, STATUS='OLD', ERR=10)
C-----------------------------------------------------------------------

      WRITE(6,27)
   27 FORMAT(/, ' Enter the tolerance = ')
      READ(5,*) TOL

C-------------------------------------------------

      WRITE(6,26)
   26 FORMAT(/, ' Enter Max. number of iteration = ')
      READ(5,*) MAXIT

C-------------------------------------------------

      WRITE(6,902)
  902 FORMAT(/,'  START TIME :')
      CALL GETTIM(tmphour, tmpminute, tmpsecond, tmphund)
      WRITE (*,900) tmphour,tmpminute,tmpsecond,tmphund
  900 FORMAT(8X,I2,':',I2.2,':',I2.2,':',I2.2)

C-----------------------------------------------------------------
C     READ COEFFICIENT FOR THE SYSTEM OF EQUATION
C-----------------------------------------------------------------

      READ(7,*) N
      DO 100 I = 1,N
      READ(7,*) (A(I,J), J=1,N), B(I)
  100 CONTINUE

C-----------------------------------------------------------------------
C     SOLVE SYSTEM OF EQUATION BY PRE-CONJUGATE GRADIENT
C-----------------------------------------------------------------------

      CALL PCG(A,B,X,TOL,MAXIT,MXNEQ,N)

C-----------------------------------------------------------------------
```

```
      WRITE(6,1001)
 1001 FORMAT(/, '  END TIME :')
      CALL GETTIM(tmphour, tmpminute, tmpsecond, tmphund)
      WRITE (*,1002) tmphour,tmpminute,tmpsecond,tmphund
 1002 FORMAT(8X,I2,':',I2.2,':',I2.2,':',I2.2)

C-----------------------------------------------------------------------

  930 WRITE(6,940)
  940 FORMAT(/, ' ENTER THE OUTPUT FILE NAME',/)
      READ(5 , '(A)', ERR=930 ) NAME2
      OPEN(UNIT=8, FILE=NAME2 , STATUS='NEW',ERR=930)

      DO 1000 I=1,N
      WRITE(8,950) I, X(I)
  950 FORMAT(I6,2X,E12.4)
 1000 CONTINUE

      STOP
      END

C-----------------------------------------------------------------------C

      SUBROUTINE PCG(A,B,X,TOL,MAXIT,MXNEQ,N)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MXNEQ,MXNEQ), B(MXNEQ), X(MXNEQ), Q(MXNEQ)
      DIMENSION R(MXNEQ), D(MXNEQ), P(MXNEQ), S(MXNEQ)
      ITER = 1

C-----------------------------------------------------------------------
C     CONSTRUCT PERCONDITION MATRIX M (BUT THIS WILL FIND M Inverse)
C-----------------------------------------------------------------------
      DO 200 I=1,N
      X(I) = 0.
      P(I) = 0.
  200 CONTINUE
      DO 210 I=1,N
      P(I) = 1./A(I,I)
  210 CONTINUE

C--------------------
C     FIND Ro
C--------------------
      DO 10 I=1,N
      SUM = 0.
      DO 20 J=1,N
      SUM = SUM + A(I,J)*X(J)
   20 CONTINUE
      R(I) = B(I) - SUM
   10 CONTINUE

C--------------------
C     FIND Do
C--------------------
      DO 30 I=1,N
      D(I) = 0.
      D(I) = R(I)*P(I)
   30 CONTINUE

C--------------------
C     FIND D-New
C--------------------
      DNEW = 0.
      DO 40 I=1,N
      DNEW = DNEW + R(I)*D(I)
   40 CONTINUE
      DEL = DNEW
```

```
    5 IF(ITER.LT.MAXIT.AND.DNEW.GT.(TOL*TOL*DEL)) THEN

C--------------------
C     FIND Q(i+1)
C--------------------
      DO 50 I=1,N
      SUM = 0.
      DO 60 J=1,N
      SUM = SUM + A(I,J)*D(J)
   60 CONTINUE
      Q(I) = SUM
   50 CONTINUE
C--------------------
C     FIND ALPHA
C--------------------
      BOTTOM = 0.
      DO 70 I=1,N
      BOTTOM = BOTTOM + D(I)*Q(I)
   70 CONTINUE
      ALPHA = DNEW/BOTTOM
      DO 80 I=1,N
      X(I) = X(I) + ALPHA*D(I)
   80 CONTINUE
C--------------------
C     FIND R(i+1)
C--------------------
      IF(ITER.EQ.50) THEN
      DO 100 I=1,N
      SUM = 0.
      DO 90 J=1,N
      SUM = SUM + A(I,J)*X(J)
   90 CONTINUE
      R(I) = B(I) - SUM
  100 CONTINUE
      ELSE
      DO 110 I=1,N
      R(I) = R(I) - ALPHA*Q(I)
  110 CONTINUE
      ENDIF
C--------------------
C     FIND S(i+1)
C--------------------
      DO 120 I=1,N
      S(I) = 0.
      S(I) = R(I)*P(I)
  120 CONTINUE
C--------------------
C     FIND BETA
C--------------------
      DOLD = DNEW
      DNEW = 0.
      DO 130 I=1,N
      DNEW = DNEW + R(I)*S(I)
  130 CONTINUE
      BETA = DNEW/DOLD
      DO 140 I=1,N
      D(I) = S(I) + BETA*D(I)
  140 CONTINUE

C-------------------------------------------------------------
      ITER = ITER + 1
      GOTO 5
      ENDIF

      RETURN
      END
```

ค.3 ลักษณะของข้อมูลที่โปรแกรม PCG ต้องการ

ลักษณะของข้อมูลที่โปรแกรมต้องการ ประกอบไปด้วยส่วนต่าง ๆดังนี้

<u>บรรทัดแรก</u>	บอกจำนวนสมการที่จะทำการแก้
<u>บรรทัดต่อ ๆไป</u>	ค่าสัมประสิทธิ์ของระบบสมการโดยเขียนบรรทัดละหนึ่งสมการ

ตัวอย่างเช่น: ถ้าจะทำการแก้ระบบสมการที่มีจำนวนสมการ 2 สมการดังนี้

$$\begin{aligned} 3x_1 + 2x_2 &= 2 \\ 2x_1 + 6x_2 &= -8 \end{aligned}$$

ลักษณะไฟล์ของข้อมูลนำเข้าจะมีลักษณะดังนี้

```
2
3     2      2
2     6     -8
```

ค.4 ตัวอย่างการใช้โปรแกรม PCG ในการแก้ระบบสมการ

ในหัวข้อนี้จะได้แสดงการแก้ระบบสมการด้วยโปรแกรม PCG โดยระบบสมการที่จะนำมาใช้นี้ประกอบไปด้วย 5 สมการ ซึ่งสามารถเขียนในรูปเมตริกซ์ได้ดังนี้

$$\begin{bmatrix} 10 & 1 & 2 & 3 & 4 \\ 1 & 9 & -1 & 2 & -3 \\ 2 & -1 & 7 & 3 & -5 \\ 3 & 2 & 3 & 12 & -1 \\ 4 & -3 & -5 & -1 & 15 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{Bmatrix} = \begin{Bmatrix} 12 \\ -27 \\ 14 \\ -17 \\ 12 \end{Bmatrix}$$

โดยที่เมตริกซ์สัมประสิทธิ์ของระบบสมการนี้มีคุณสมบัติสมมาตร และ positive definite จึงสามารถนำโปรแกรม PCG มาใช้ในการแก้ระบบสมการนี้ได้ ซึ่งคุณสมบัติของเมตริกซ์เหล่านี้สามารถพบได้ในสมการลาปลาซ (Laplace's equation) หรือพบได้ในปัญหาการนำความร้อน (heat conduction problem) นั่นเอง ซึ่งจะเห็นได้ว่าลักษณะของสมการสำหรับความดันดังแสดงในสมการ (4.22) ในบทที่ 4 นั้นมีลักษณะของสมการลาปลาซอยู่จึงสามารถใช้ระเบียบวิธีนี้ในการแก้ระบบสมการดังกล่าวได้

สำหรับข้อมูลที่จะใช้ในการแก้ระบบสมการนี้สมมติว่าบรรจุอยู่ในไฟล์ชื่อ 'EX2.DAT' ซึ่งมีรายละเอียดดังนี้

```
 5
10      1      2      3      4      12
 1      9     -1      2     -3     -27
 2     -1      7      3     -5      14
 3      2      3     12     -1     -17
 4     -3     -5     -1     15      12
```

รูปที่ ค.2 ข้อมูลในไฟล์ชื่อ 'EX2.DAT'

เมื่อผู้ใช้ได้ใช้โปรแกรม PCG ในการแก้ระบบสมการ โปรแกรมจะถามชื่อไฟล์ข้อมูลซึ่งผู้ใช้จะพิมพ์ตอบกลับไป จากนั้นโปรแกรมจะให้ป้อนค่าของค่าความผิดพลาดที่จะยอมให้ได้ และจำนวนรอบสูงสุดของการคำนวณแบบทำซ้ำ และโปรแกรมจะเริ่มการทำงานดังที่ได้อธิบายข้างต้นในหัวข้อ ค.1 เมื่อการคำนวณสิ้นสุดลง โปรแกรมจะให้ใส่ชื่อไฟล์ที่จะบรรจุผลคำตอบที่คำนวณได้ โดยขั้นตอนทั้งหมดดังกล่าวมานี้จะปรากฏบนจอคอมพิวเตอร์ดังแสดงในรูปที่ ค.3

```
>PCG        <Enter>

PLEASE ENTER INPUT FILE NAME:
EX2.DAT

Enter the tolerance =
1.E-10

Enter Max. number of iteration =
200

 START TIME :
  10/08/2000        23:48:38:09

 END TIME :
  10/08/2000        23:48:38:09

ENTER THE OUTPUT FILE NAME
EX2.OUT
```

รูปที่ ค.3 ลำดับขั้นตอนที่ปรากฏบนจอคอมพิวเตอร์ในขณะใช้โปรแกรม PCG

ผลลัพธ์ของการแก้ระบบสมการนี้ ซึ่งบรรจุอยู่ในไฟล์ชื่อ 'EX2.OUT' ได้แสดงในรูป ค.4

```
1     .1000E+01
2    -.2000E+01
3     .3000E+01
4    -.2000E+01
5     .1000E+01
```

รูปที่ ค.4 ผลลัพธ์ที่คำนวณได้จากโปรแกรม PCG

ซึ่งผลลัพธ์ดังกล่าวเมื่อนำมาตรวจสอบกับระบบสมการแล้วปรากฏว่าได้ค่าที่ถูกต้องแม่นยำ

ประวัติผู้เขียนวิทยานิพนธ์

นายนิพนธ์ วรรณโสภาคย์ เกิดเมื่อวันที่ 31 เดือนพฤษภาคม พุทธศักราช 2518 จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาปริญญาวิศวกรรมศาสตรบัณฑิตจากภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2539 เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2540