```
!*************************************************************************
**************************************
!*************************************************************************
**************************************
!****
                                    *****
!****          MAIN PROGRAM
                                    *****
!****
                                    *****
!****-------------------------------------------------------------------------
-----------------------------------*****
!****
                                    *****
!****    THIS PROGRAM CALCULATES THE AMOUNT OF RADIATION INDUCED SEGREGATION
FOR A TERNARY CONCENTRATED ALLOY.     *****
!**** THE FORMULATION IS BASED ON THE PERKS MODEL AND IS SOLVED NUMERICALLY
USING THE GEAR SUBROUTINES.            *****
!****-------------------------------------------------------------------------
-----------------------------------*****
!**** This version is a rewrite of the original and is designed to conform to
Fortran 95 standards.  It also       *****
!**** set the format of the program to a single style.  Below is a listing of
changes to the original program other*****
!**** than that absolutly required by changes in Fortran from Fortran 2 through
95.                                 *****
!****
                                    *****
!**** Changes:
                                    *****
!****         STEP changed to ISTEP
                                    *****
!****         STOP changed to PSTOP   you just can't use a reserved word as a
variable                             *****
!****         INDEX changed to IERR   you just can't use a reserved word as a
variable                             *****
!****         In the original files were opened in the begining of the program
and closed at the end - no longer   *****
!****
                                    *****
!*************************************************************************
**************************************
      MODULE MOD_ALL

      PARAMETER (IP = 5000) ! modified from (IP = 500)

      !CONCENTRATIONS
      REAL*8  :: CA(IP),CB(IP),CC(IP),CV(IP),CI(IP)
      REAL*8  :: NA(IP),NB(IP),NC(IP),NV(IP),NI(IP)

      !DIFFUSIVITIES
      REAL*8  :: DAV(IP),DBV(IP),DCV(IP),DAIO(IP),DBIO(IP),DCIO(IP),AL
```

```
      !GEOMETRY
      REAL*8  :: MESHSP(IP),NAT
      REAL*8  :: XVALUE(IP)
      REAL*8  :: MESHSI(IP)

      !FLUXES
      REAL*8  :: JA(IP),JB(IP),JC(IP),JV(IP),JI(IP),JA0,JB0,JC0

      !DEFECTS
      REAL*8  ::
RECA(IP),RECB(IP),RECC(IP),DISLOC(IP),CVTHER(IP),DISPV(IP),DISPI(IP)
      REAL*8  :: DIFI(IP),DIFV(IP),BIASI,BIASV
      REAL*8  :: TKT(IP)

      !TIME
      REAL*8  :: TOUTPT(IP),TSTOP
      REAL*8  :: TOUT

      !DAMAGE
      REAL*8  :: DISPRT

      !ENERGIES
      REAL*8  :: EAA,EBB,ECC,Z
      REAL*8  :: EAB,EBC,EAC
      REAL*8  :: EAV,EBV,ECV
      REAL*8  :: ESA,ESB,ESC
      REAL*8  :: PREVA,PREVB,PREVC,LAMBDA

      INTEGER :: NSTEP,NOUT


      END MODULE MOD_ALL
!
      MODULE MOD_SOLVER

      REAL*8, ALLOCATABLE  :: Y0(:)
      REAL*8, ALLOCATABLE  :: Y(:)
      REAL*8, ALLOCATABLE  :: YDOT(:)
      REAL*8, ALLOCATABLE  :: RWORK(:)
      INTEGER, ALLOCATABLE :: IWORK(:)
      INTEGER :: N,MF,IERR,ITOL,IOPT,ITASK,ISTEP
      REAL*8  :: EPS,EPSA
      CHARACTER :: PSTOP

      END MODULE MOD_SOLVER
!
      PROGRAM MAIN
!
!     Program Initialization Begins Here
!     ==================================
!     Variable Initialization
```

```
!     ----------------------
      USE MOD_ALL
      USE MOD_SOLVER

      IMPLICIT REAL*8 (A-H,O-Z)
!
!     Format Statements
!     -----------------
  100 FORMAT ("ERROR RETURN WITH IERR= ",I3)
!
!
!     Program Execution Starts Here
!     =============================
      ISTEP=0
      PSTOP='N'
      CALL INITIALIZE
!
      DO WHILE(PSTOP .EQ. 'N')
         CALL PREP
         IF(PSTOP.EQ.'Y') exit
         CALL FEX(N,T,Y,YDOT)
         CALL
DLSODES(FEX,N,Y,T,TOUT,ITOL,EPS,EPSA,ITASK,IERR,IOPT,RWORK,N**2+20,IWORK,2*N+20
,JEX,MF)
         IF (IERR.LT.-1) THEN
            OPEN(UNIT=8,FILE='perks.err',STATUS='UNKNOWN')
            WRITE (8,100) IERR
            CLOSE(8)
            print *,  "terminated with error"
            go to 999
         ELSEIF (IERR.EQ.-1) THEN
            IERR=2
         END IF
         CALL OUTPUT
      END DO
!
      print *,  "Completed"
 999  close (6)
      close (7)
      STOP
!
!     Program Completed
!     =================
      END PROGRAM MAIN
!
      subroutine INITIALIZE
!
!     Subroutine initialization starts here
!     =====================================
!     Variables
!     ---------
      USE MOD_ALL
```

```
      USE MOD_SOLVER
      IMPLICIT REAL*8 (A-H,O-Z)
!
      DIMENSION DFRAC(IP),TFRAC(IP),SFRAC(IP)
      DIMENSION CAFRAC(IP),CBFRAC(IP),CCFRAC(IP)
      DIMENSION TEMP(IP)
      DIMENSION NUIA(IP),NUIB(IP),NUIC(IP)
!
      REAL*8 :: NUOV,NUOI
      REAL*8 :: NUIA,NUIB,NUIC
      CHARACTER (KIND=1) :: FRAC
      character fn*80
      logical flag
!     Format Statements
!     -----------------
 1000 FORMAT(3F8.1,3I3)
 1100 FORMAT(2E12.4)
 1200 FORMAT(E19.8,2F11.8,F6.1)
 1300 FORMAT(F12.8)
 1400 FORMAT(2F10.7)
 1500 FORMAT(3E16.5)
 1600 FORMAT(4F6.3)
 1700 FORMAT(3F11.8)
 1800 FORMAT(3F11.8)
 1850 FORMAT(4F11.8)
 1900 FORMAT(2E12.4)
 2000 FORMAT(F11.8,F6.2,2F5.2)
 2100 FORMAT(6(E12.4))
 2200 FORMAT(A1)
 2300 FORMAT(3F4.2)
 2310 FORMAT(4F11.8)
 2312 FORMAT(3F11.8)
 2315 FORMAT(3F11.8)
 2320 FORMAT(6(F7.4))
 2321 FORMAT(6(F4.1))
 2322 FORMAT(6(F4.1))
 2323 FORMAT(6(F4.1))
 2324 FORMAT(6(F4.1))
 2325 FORMAT(6(F4.1))
 2331 FORMAT(1X,"TFRAC=",/,6(F7.4))
 2332 FORMAT(1X,"CAFRAC=",/,6(F7.4))
 2333 FORMAT(1X,"CBFRAC=",/,6(F7.4))
 2334 FORMAT(1X,"CCFRAC=",/,6(F7.4))
 2335 FORMAT(1X,"DFRAC=",/,6(F7.4))
 2336 FORMAT(1X,"SFRAC=",/,6(F7.4))
 2400 FORMAT(1X,"EPS=",E12.4)
 2500
FORMAT(1X,"DISPRT=",E19.9,2X,"ETAV=",F11.8,2X,"ETAI=",F11.8,2X,"DOSE=",F6.1)
 2600 FORMAT(1X,"TEMP=",F12.8,"C")
 2700 FORMAT(1X,"CB=",F10.7,2X,"CC=",F10.7)
 2800 FORMAT(1X,"DISL=",E17.8,2X,"NAT=",E17.8,2X,"LAMBDA=",E17.8)
 2900 FORMAT(1X,"FAV=",F6.3,2X,"FBV=",F6.3,2X,"FCV=",F6.3,2X,"FI=",F6.3)
```

```
 3000 FORMAT(1X,"WAV=",F11.8,2X,"WBV=",F11.8,2X,"WCV=",F11.8)
 3050 FORMAT(1X,"WAI=",F11.8,2X,"WBI=",F11.8,2X,"WCI=",F11.8)
 3100 FORMAT(1X,"ECOHA=",F11.8,2X,"ECOHB=",F11.8,2X,"ECOHC=",F11.8)
 3150 FORMAT(1X,"EMIA=",F11.8,2X,"EMIB=",F11.8,2X,"EMIC=",F11.8,2X,"SV=",F11.8)
 3200 FORMAT(1X,"NUOV",E12.4,2X,"NUOI=",E12.4)
 3300 FORMAT(1X,"AL=",F11.8,2X,"Z=",F6.2,1X,"BIASV=",F5.2,1X,"BIASI=",F5.2)
 3310 FORMAT(1X,"EFA=",F11.8,1X,"EFB=",F11.8,1X,"EFC=",F11.8,1X,"EFGB=",F11.8)
 3312 FORMAT(1X,"EMA=",F11.8,1X,"EMB=",F11.8,1X,"EMC=",F11.8)
 3315 FORMAT(1X,"EORDAB=",F11.8,1X,"EORDAC=",F11.8,1X,"EORDBC=",F11.8)
!
!     Subroutine Execution Starts Here
!     ================================
! input file
      open(UNIT=5,FILE='./perks.in',STATUS='OLD')
! ouput file
      open(UNIT=6,FILE='./perks.out',STATUS='REPLACE')
!
      READ(5,*) R1,R2,RF,N1,N2,N3
!
      READ(5,*) EPS,EPSA
      READ(5,*) DISPRT,ETAV,ETAI,DOSE
      READ(5,*) TEMPC
      READ(5,*) CONCB,CONCC
      READ(5,*) DISL,NAT,LAMBDA
      READ(5,*) FAV,FBV,FCV,FI
      READ(5,*) WAV,WBV,WCV
      READ(5,*) WAI,WBI,WCI
!           *
      READ(5,*) ECOHA,ECOHB,ECOHC
      READ(5,*) EMIA,EMIB,EMIC,SV
      READ(5,*) EMA,EMB,EMC
      READ(5,*) EFA,EFB,EFC,EFGB
      READ(5,*) EORDAB,EORDAC,EORDBC
      READ(5,*) NUOV,NUOI
      READ(5,*) AL,Z,BIASV,BIASI
!
      READ(5,*) NOUT,(TOUTPT(I),I=1,NOUT)
      READ(5,*) FRAC
!
      NSTEP=N1+N2+N3
      N=NSTEP*5
      ALLOCATE(Y0(N),Y(N),YDOT(N),RWORK(N**2+20),IWORK(2*N+20))
!
      IF(FRAC.EQ.'N') THEN
         DO I=1,NSTEP-1
            TFRAC(I)=1.0
         END DO
!
         DO I=1,NSTEP
            CAFRAC(I)=1.0
            CBFRAC(I)=1.0
            SFRAC(I)=1.0
```

```
            DFRAC(I)=1.0
            CCFRAC(I)=1.0
          END DO
!
        ELSE
          READ(5,2320) (TFRAC(I),I=1,NSTEP-1)
          READ(5,2320) (CAFRAC(I),I=1,NSTEP)
          READ(5,2320) (CBFRAC(I),I=1,NSTEP)
          READ(5,2320) (CCFRAC(I),I=1,NSTEP)
          READ(5,2320) (DFRAC(I),I=1,NSTEP)
          READ(5,2320) (SFRAC(I),I=1,NSTEP)
        END IF

        write(6,*) 'read input'
!
        WRITE(6,2400) EPS
        WRITE(6,2500) DISPRT,ETAV,ETAI,DOSE
        WRITE(6,2600) TEMPC
        WRITE(6,2700) CONCB,CONCC
        WRITE(6,2800) DISL,NAT,LAMBDA
        WRITE(6,2900) FAV,FBV,FCV,FI
        WRITE(6,3000) WAV,WBV,WCV
        WRITE(6,3050) WAI,WBI,WCI
!
        WRITE(6,3100) ECOHA,ECOHB,ECOHC
        WRITE(6,3150) EMIA,EMIB,EMIC,SV
        WRITE(6,3312) EMA,EMB,EMC
        WRITE(6,3310) EFA,EFB,EFC,EFGB
        WRITE(6,3315) EORDAB,EORDAC,EORDBC
        WRITE(6,3200) NUOV,NUOI
        WRITE(6,3300) AL,Z,BIASV,BIASI
!
        WRITE(6,2331) (TFRAC(I),I=1,NSTEP-1)
        WRITE(6,2332) (CAFRAC(I),I=1,NSTEP)
        WRITE(6,2333) (CBFRAC(I),I=1,NSTEP)
        WRITE(6,2334) (CCFRAC(I),I=1,NSTEP)
        WRITE(6,2335) (DFRAC(I),I=1,NSTEP)
        WRITE(6,2336) (SFRAC(I),I=1,NSTEP)
!
        SCFAC=1.0E-09
        BOLTZ=8.617E-05
        TSTOP=DOSE/DISPRT
        EAA=ECOHA/(Z/2)
        EBB=ECOHB/(Z/2)
        ECC=ECOHC/(Z/2)
        EAB=0.5*(EAA+EBB)-EORDAB
        EAC=0.5*(EAA+ECC)-EORDAC
        EBC=0.5*(EBB+ECC)-EORDBC
        EAV=(ECOHA+EFA)/Z
        EBV=(ECOHB+EFB)/Z
        ECV=(ECOHC+EFC)/Z
        ESA=EMA+Z*(EAA+EAV)
```

```
      ESB=EMB+Z*(EBB+EBV)
      ESC=EMC+Z*(ECC+ECV)
      PREVA=NUOV*WAV*FAV
      PREVB=NUOV*WBV*FBV
      PREVC=NUOV*WCV*FCV
!
      CONCA=1.0-(CONCB+CONCC)
      do I=1,N1-1
         MESHSP(I)=R1*SCFAC/N1
      end do
!
      do I=N1,N1+N2-1
         MESHSP(I)=(R2-R1)*SCFAC/N2
      end do
!
      do I=N1+N2,N1+N2+N3-1
         MESHSP(I)=(RF-R2)*SCFAC/N3
      end do
!
      do I=1,NSTEP
         DISPV(I)=DISPRT*ETAV*DFRAC(I)
         DISPI(I)=DISPRT*ETAI*DFRAC(I)
      end do
!
         XVALUE(1)=0.0
      do I=2,NSTEP
         XVALUE(I)=XVALUE(I-1)+MESHSP(I-1)
      end do
!
      do I=1,NSTEP
         CA(I)=CONCA*CAFRAC(I)
         CB(I)=CONCB*CBFRAC(I)
         CC(I)=CONCC*CCFRAC(I)
         CI(I)=0.0
         DISLOC(I)=DISL*SFRAC(I)
      end do
!
!     EFV=CA(1)*EFA+CB(1)*EFB+CC(1)*EFC  commented out in original program
!
      do I=1,NSTEP-1
         TEMP(I)=(TEMPC+273)*TFRAC(I)
         TKT(I)=BOLTZ*TEMP(I)
         NUIA(I)=NUOI*WAI*FI*EXP((-1*EMIA)/TKT(I))
         NUIB(I)=NUOI*WBI*FI*EXP((-1*EMIB)/TKT(I))
         NUIC(I)=NUOI*WCI*FI*EXP((-1*EMIC)/TKT(I))
         DAIO(I)=0.66667*NUIA(I)*LAMBDA**2
         DBIO(I)=0.66667*NUIB(I)*LAMBDA**2
         DCIO(I)=0.66667*NUIC(I)*LAMBDA**2
         CVTHER(I)=EXP(SV)*EXP((-1*EFGB)/TKT(I))
      end do
      CVTHER(NSTEP)=CVTHER(NSTEP-1)
      do I=2,NSTEP-1
```

```
         CVTHER(I)=0.5*(CVTHER(I)+CVTHER(I-1))
      end do
!
      do I=1,NSTEP
         CV(I)=CVTHER(I)
      end do
!
      do I=1,NSTEP
         Y0(I)=CA(I)
      end do
      do I=NSTEP+1,2*NSTEP
         Y0(I)=CB(I-NSTEP)
      end do
      do I=2*NSTEP+1,3*NSTEP
         Y0(I)=CC(I-2*NSTEP)
      end do
      do I=3*NSTEP+1,4*NSTEP
         Y0(I)=CV(I-3*NSTEP)
      end do
      do I=4*NSTEP+1,5*NSTEP
         Y0(I)=CI(I-4*NSTEP)
      end do
!
      return
      end subroutine INITIALIZE
!
      subroutine PREP
!
!     Subroutine initialization starts here
!     =====================================
!     Variables
!     ---------
      USE MOD_ALL
      USE MOD_SOLVER
!
!     Subroutine Execution Starts Here
!     ================================
      if (ISTEP .EQ. 0) then
         N=5*NSTEP
         T0=0
         MF=222
         IERR=1
         TOUT=TOUTPT(1)
         ISTEP=ISTEP+1
         ITOL=1
         ITASK=1
         IOPT=1
         Y=Y0
         IWORK(6)=1000000
      else if (ISTEP.LT.NOUT) then
         ISTEP=ISTEP+1
         TOUT=TOUTPT(ISTEP)
```

```
      else
         PSTOP='Y'
         write(6,*) "Stopping Time Reached"
      end if
!
!     Subroutine completed - Time to return
!     =====================================
      return
      end subroutine Prep
!
      subroutine OUTPUT
!
!     Subroutine initialization starts here
!     =====================================
!     Variables
!     ---------
      USE MOD_ALL
      USE MOD_SOLVER

      IMPLICIT REAL*8 (A-H,O-Z)
!
      DIMENSION CERR(IP)
!
      DIMENSION XOUT(IP)
!
!     Format Statements
!     -----------------
  100 FORMAT (/1X,"TIME=",E8.1,2X,"DOSE=",F8.2)
  110 FORMAT (/7X,"POSITION",11X," CA",17X," CB",17X," CC",17X," CV",17x," CI")
  115 FORMAT (/1X)
  120 FORMAT (1X,6(E16.8,4X))
  130
FORMAT(/10X,"CASURF=",4x,F8.4,21X,"CBSURF=",6x,F8.4,19X,"CCSURF=",6x,F8.4)
!
!    Subroutine Execution Start Here
!    ===============================
      do I=1,NSTEP
         CA(I)=Y(I)
      end do
      do I=NSTEP+1,2*NSTEP
         CB(I-NSTEP)=Y(I)
      end do
      do I=2*NSTEP+1,3*NSTEP
         CC(I-2*NSTEP)=Y(I)
      end do
      do I=3*NSTEP+1,4*NSTEP
         CV(I-3*NSTEP)=Y(I)
      end do
      do I=4*NSTEP+1,5*NSTEP
         CI(I-4*NSTEP)=Y(I)
      end do
!
```

```
      DOSE=DISPRT*TOUTPT(ISTEP)
!
!      open(UNIT=6,FILE='perks.out',STATUS='OLD',POSITION='APPEND')
      WRITE (6,'(/1X,"TIME=",E8.1,2X,"DOSE=",e10.4)') TOUTPT(ISTEP),DOSE
      print *,"TOUTPT,DOSE",TOUTPT(ISTEP),DOSE
      WRITE (6,110)
      do I=1,NSTEP
         CERR(I)=1-(CA(I)+CB(I)+CC(I))
         XOUT(I)=XVALUE(I)*1E9
      WRITE (6,120) XOUT(I),CA(I),CB(I),CC(I),CV(I),CI(I)
      end do
!
      SUM1=0
      SUM2=0
      do J=1,NSTEP
         SUM1=SUM1+EXP((-XOUT(J)/.8452))
         SUM2=SUM2+CA(J)*EXP((-XOUT(J)/.8452))
      end do
      CASURF= SUM2/SUM1
!
      SUM1=0
      SUM2=0
      do J=1,NSTEP
         SUM1=SUM1+EXP((-XOUT(J)/.7474))
         SUM2=SUM2+CB(J)*EXP((-XOUT(J)/.7474))
      end do
      CBSURF= SUM2/SUM1
!
!
      SUM1=0
      SUM2=0
      do J=1,NSTEP
         SUM1=SUM1+EXP((-XOUT(J)/.9472))
         SUM2=SUM2+CC(J)*EXP((-XOUT(J)/.9472))
      end do
      CCSURF= SUM2/SUM1
      TEMP1=CASURF
      TEMP2=CBSURF
      TEMP3=CCSURF
      CASURF=TEMP1/(TEMP1+TEMP2+TEMP3)
      CBSURF=TEMP2/(TEMP1+TEMP2+TEMP3)
      CCSURF=TEMP3/(TEMP1+TEMP2+TEMP3)
!
      WRITE(6,130) CASURF,CBSURF,CCSURF
!      close(6)
!
!     Subroutine completed - Time to return
!     ====================================
!     write(7,*)   "outpt ok"
      return
      end subroutine Output
!
```

```
      subroutine FEX (N,T,Y,YDOT)
!
!     Subroutine initialization starts here
!     =====================================
!     Variables
!     ---------
!     Format Statements
!     -----------------
      USE MOD_ALL
      IMPLICIT REAL*8 (A-H,O-Z)
!
      DIMENSION RECOMB(IP),INTSINK(IP),VACSINK(IP),VACSOUR(IP)
      DIMENSION DIVJA(IP),DIVJB(IP),DIVJC(IP),DIVJV(IP),DIVJI(IP)
      DIMENSION YDOT(N),Y(N)
      DIMENSION CADOT(IP),CBDOT(IP),CCDOT(IP),CVDOT(IP),CIDOT(IP)
      DIMENSION GRADCA(IP),GRADCB(IP),GRADCC(IP),GRADCV(IP),GRADCI(IP)
      DIMENSION DA(IP),DB(IP),DC(IP),DV(IP),DI(IP)
      DIMENSION JO(IP),DAI(IP),DBI(IP),DCI(IP)
      DIMENSION EA(IP),EB(IP),EC(IP)
      DIMENSION NUVA(IP),NUVB(IP)
      DIMENSION NUVC(IP),NUIA(IP),NUIB(IP),NUIC(IP)
!
      REAL*8 :: JO,INTSINK
      REAL*8 :: NUVA,NUVB,NUVC
      REAL*8 :: NUIA,NUIB,NUIC
!
      do I=1,NSTEP
         CA(I)=Y(I)
      end do
      do I=NSTEP+1,2*NSTEP
         CB(I-NSTEP)=Y(I)
      end do
      do I=2*NSTEP+1,3*NSTEP
         CC(I-2*NSTEP)=Y(I)
      end do
      do I=3*NSTEP+1,4*NSTEP
         CV(I-3*NSTEP)=Y(I)
      end do
      do I=4*NSTEP+1,5*NSTEP
         CI(I-4*NSTEP)=Y(I)
      end do
      do I=1,NSTEP-1
         NA(I)=0.5*(CA(I+1)+CA(I))
         NB(I)=0.5*(CB(I+1)+CB(I))
         NC(I)=0.5*(CC(I+1)+CC(I))
         NV(I)=0.5*(CV(I+1)+CV(I))
         NI(I)=0.5*(CI(I+1)+CI(I))
         DAI(I)=DAIO(I)
         DBI(I)=DBIO(I)
         DCI(I)=DCIO(I)

EA(I)=((ESA+ESA*NA(I)+ESB*NB(I)+ESC*NC(I))/2)-((Z*(NA(I)*EAA+NB(I)*EAB+NC(I)*EA
```

```
C+NV(I)*EAV)) &
              +(Z*(NA(I)*EAV+NB(I)*EBV+NC(I)*ECV)))

EB(I)=((ESB+ESA*NA(I)+ESB*NB(I)+ESC*NC(I))/2)-((Z*(NA(I)*EAB+NB(I)*EBB+NC(I)*EB
C+NV(I)*EBV)) &
              +(Z*(NA(I)*EAV+NB(I)*EBV+NC(I)*ECV)))

EC(I)=((ESC+ESA*NA(I)+ESB*NB(I)+ESC*NC(I))/2)-((Z*(NA(I)*EAC+NB(I)*EBC+NC(I)*EC
C+NV(I)*ECV)) &
              +(Z*(NA(I)*EAV+NB(I)*EBV+NC(I)*ECV)))
         NUVA(I)=PREVA*EXP((-1*EA(I)/TKT(I)))
         NUVB(I)=PREVB*EXP((-1*EB(I)/TKT(I)))
         NUVC(I)=PREVC*EXP((-1*EC(I)/TKT(I)))
         DAV(I)=NUVA(I)*LAMBDA**2
         DBV(I)=NUVB(I)*LAMBDA**2
         DCV(I)=NUVC(I)*LAMBDA**2
         RECA(I)=(NUVA(I)+NUIA(I))*Z
         RECB(I)=(NUVB(I)+NUIB(I))*Z
         RECC(I)=(NUVC(I)+NUIC(I))*Z
         DIFV(I)=DAV(I)*NA(I)+DBV(I)*NB(I)+DCV(I)*NC(I)
         DIFI(I)=DAI(I)*NA(I)+DBI(I)*NB(I)+DCI(I)*NC(I)
      end do
!
      DIFV(NSTEP)=DIFV(NSTEP-1)
      DIFI(NSTEP)=DIFI(NSTEP-1)
      do I=2,NSTEP-1
         DIFV(I)=0.5*(DIFV(I)+DIFV(I-1))
         DIFI(I)=0.5*(DIFI(I)+DIFI(I-1))
      end do
!
      RECA(NSTEP)=RECA(NSTEP-1)
      RECB(NSTEP)=RECB(NSTEP-1)
      RECC(NSTEP)=RECC(NSTEP-1)
      CVTHER(NSTEP)=CVTHER(NSTEP-1)
      do I=2,NSTEP-1
         RECA(I)=0.5*(RECA(I)+RECA(I-1))
         RECB(I)=0.5*(RECB(I)+RECB(I-1))
         RECC(I)=0.5*(RECC(I)+RECC(I-1))
         CVTHER(I)=0.5*(CVTHER(I)+CVTHER(I-1))
      end do
!
      JA0=0.0
      JB0=0.0
      JC0=0.0
      JA(NSTEP)=0.0
      JB(NSTEP)=0.0
      JC(NSTEP)=0.0
      JV(NSTEP)=0.0
      JI(NSTEP)=0.0
!
      do I=1,NSTEP-1
!
```

```
         GRADCA(I)=(CA(I+1)-CA(I))/MESHSP(I)
         GRADCB(I)=(CB(I+1)-CB(I))/MESHSP(I)
         GRADCC(I)=(CC(I+1)-CC(I))/MESHSP(I)
         GRADCV(I)=(CV(I+1)-CV(I))/MESHSP(I)
         GRADCI(I)=(CI(I+1)-CI(I))/MESHSP(I)
!
         DA(I)=DAV(I)*NV(I)+DAI(I)*NI(I)
         DB(I)=DBV(I)*NV(I)+DBI(I)*NI(I)
         DC(I)=DCV(I)*NV(I)+DCI(I)*NI(I)
         DV(I)=DAV(I)*NA(I)+DBV(I)*NB(I)+DCV(I)*NC(I)
         DI(I)=DAI(I)*NA(I)+DBI(I)*NB(I)+DCI(I)*NC(I)
!
      end do
!
      do I=1,NSTEP-1

JV(I)=NAT*(-1*DV(I)*GRADCV(I)+NV(I)*AL*(DAV(I)*GRADCA(I)+DBV(I)*GRADCB(I)+DCV(I
)*GRADCC(I)))

JI(I)=NAT*(-1*DI(I)*GRADCI(I)-NI(I)*AL*(DAI(I)*GRADCA(I)+DBI(I)*GRADCB(I)+DCI(I
)*GRADCC(I)))
         JO(I)=JI(I)-JV(I)

JA(I)=NAT*(-1*DA(I)*AL*GRADCA(I)+NA(I)*(DAV(I)*GRADCV(I)-DAI(I)*GRADCI(I)))-JO(
I)*NA(I)

JB(I)=NAT*(-1*DB(I)*AL*GRADCB(I)+NB(I)*(DBV(I)*GRADCV(I)-DBI(I)*GRADCI(I)))-JO(
I)*NB(I)

JC(I)=NAT*(-1*DC(I)*AL*GRADCC(I)+NC(I)*(DCV(I)*GRADCV(I)-DCI(I)*GRADCI(I)))-JO(
I)*NC(I)
!
         JV(I)=JV(I)-JO(I)*NV(I)
         JI(I)=JI(I)-JO(I)*NI(I)
      end do
!
      DIVJA(1)=2.0*(JA(1)-JA0)/MESHSP(1)
      DIVJB(1)=2.0*(JB(1)-JB0)/MESHSP(1)
      DIVJC(1)=2.0*(JC(1)-JC0)/MESHSP(1)
!
      do I=2,NSTEP-1
         MESHSI(I)=0.5*(MESHSP(I)+MESHSP(I-1))
         DIVJA(I)=(JA(I)-JA(I-1))/MESHSI(I)
         DIVJB(I)=(JB(I)-JB(I-1))/MESHSI(I)
         DIVJC(I)=(JC(I)-JC(I-1))/MESHSI(I)
         DIVJV(I)=(JV(I)-JV(I-1))/MESHSI(I)
         DIVJI(I)=(JI(I)-JI(I-1))/MESHSI(I)
      end do
!
      DIVJA(NSTEP)=2.0*(JA(NSTEP)-JA(NSTEP-1))/MESHSP(NSTEP-1)
      DIVJB(NSTEP)=2.0*(JB(NSTEP)-JB(NSTEP-1))/MESHSP(NSTEP-1)
      DIVJC(NSTEP)=2.0*(JC(NSTEP)-JC(NSTEP-1))/MESHSP(NSTEP-1)
```

```
      DIVJV(NSTEP)=2.0*(JV(NSTEP)-JV(NSTEP-1))/MESHSP(NSTEP-1)
      DIVJI(NSTEP)=2.0*(JI(NSTEP)-JI(NSTEP-1))/MESHSP(NSTEP-1)
!
      do I=1,NSTEP
         CADOT(I)=-1*DIVJA(I)/NAT
         CBDOT(I)=-1*DIVJB(I)/NAT
         CCDOT(I)=-1*DIVJC(I)/NAT
      end do
!
      do I=1,NSTEP
         RECOMB(I)=RECA(I)*CA(I)+RECB(I)*CB(I)+RECC(I)*CC(I)
         INTSINK(I)=DISLOC(I)*DIFI(I)
         VACSINK(I)=DISLOC(I)*DIFV(I)
         VACSOUR(I)=DISLOC(I)*DIFV(I)*CVTHER(I)
      end do
!
      CVDOT(1)=0.0
      CIDOT(1)=0.0
      do I=2,NSTEP

CVDOT(I)=-1*DIVJV(I)/NAT-RECOMB(I)*CV(I)*CI(I)-BIASV*VACSINK(I)*CV(I)+VACSOUR(I
)+DISPV(I)

CIDOT(I)=-1*DIVJI(I)/NAT-RECOMB(I)*CV(I)*CI(I)-BIASI*INTSINK(I)*CI(I)+DISPI(I)
      end do
!
      do I=1,NSTEP
         Y(I)=CA(I)
      end do
!
      do I=NSTEP+1,2*NSTEP
         Y(I)=CB(I-NSTEP)
      end do
!
      do I=2*NSTEP+1,3*NSTEP
         Y(I)=CC(I-2*NSTEP)
      end do
!
      do I=3*NSTEP+1,4*NSTEP
         Y(I)=CV(I-3*NSTEP)
      end do
!
      do I=4*NSTEP+1,5*NSTEP
         Y(I)=CI(I-4*NSTEP)
      end do
!
      do I=1,NSTEP
         YDOT(I)=CADOT(I)
      end do
!
      do I=NSTEP+1,2*NSTEP
         YDOT(I)=CBDOT(I-NSTEP)
```

```
      end do
!
      do I=2*NSTEP+1,3*NSTEP
         YDOT(I)=CCDOT(I-2*NSTEP)
      end do
!
      do I=3*NSTEP+1,4*NSTEP
         YDOT(I)=CVDOT(I-3*NSTEP)
      end do
!
      do I=4*NSTEP+1,5*NSTEP
         YDOT(I)=CIDOT(I-4*NSTEP)
      end do
!
!     Subroutine completed - Time to return
!     =====================================
!!     write(7,*)   "diffun ok"
      return
      end subroutine FEX
!
      subroutine JEX (N,T,Y,ML,MU,PD,NRPD)
!-----------------------------------------------------------------------

      implicit real*8 (a-h,o-z)
      dimension Y(N),PD(NRPD,N)

      return
      end
```