```
program vect3d

!aim:: to use the fourth order RK method to solve a 3d heat equation for the
!vector u=(u1,u2,u3).

implicit none
include "vect3dparam"

doubleprecision::u(1:3,0:n-1,0:n-1,0:n-1),v1(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::v2(1:3,0:n-1,0:n-1,0:n-1),v3(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::v4(1:3,0:n-1,0:n-1,0:n-1),x(0:n),y(0:n),z(0:n)
doubleprecision::k1(1:3,0:n-1,0:n-1,0:n-1),k2(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::k3(1:3,0:n-1,0:n-1,0:n-1),k4(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::f1(1:3,0:n-1,0:n-1,0:n-1),f2(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::f3(1:3,0:n-1,0:n-1,0:n-1),f4(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::x1,x2,pi,h,t
integer::i,j,k,l

x1=0.0
x2=1.0
pi=4.0*atan(1.0)
h=(x2-x1)/n

t=0.0
i=0

do j=0,n-1
   x(j)=j*h

      do k=0,n-1
         y(k)=k*h

            do l=0,n-1
               z(l)=l*h

!DEFINE THE INITIAL CONDITIONS, t=0.0
            u(1,j,k,l)=sin(2.0*pi*x(j))+cos(2.0*pi*y(k))+cos(2.0*pi*z(l))
            u(2,j,k,l)=cos(2.0*pi*y(k))
            u(3,j,k,l)=cos(2.0*pi*z(l))

            end do

      end do

end do

x(n)=1.0
y(n)=1.0
z(n)=1.0

call plot(t,i,j,k,l,x,y,z,u)

!BEGIN TIME LOOP
do i=1,nsteps

               v1=u

               call sub1(v1,t,f1)
               k1=dt*f1
               v2=v1+0.5*k1
               t=t+0.5*dt
```



```
               call sub1(v2,t,f2)
               k2=dt*f2
               v3=v1+0.5*k2

               call sub1(v3,t,f3)
               k3=dt*f3
               v4=v1+k3
               t=t+0.5*dt

               call sub1(v4,t,f4)
               k4=dt*f4

               u=v1+(1.0/6.0)*(k1+(2*k2)+(2*k3)+k4)

         if(mod(i,100)==0)call plot(t,i,j,k,l,x,y,z,u)

end do
end program vect3d

!**********************************************************************

subroutine sub1(v,t,f)
implicit none
include "vect3dparam"
doubleprecision::v(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::f(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::t
real::h=1.0/n


integer::c,j,jplus,jminus
integer::k,kplus,kminus
integer::l,lplus,lminus

do j=0,n-1

       if(j==0)jminus=n-1
       if(j>0)jminus=j-1
       if(j==n-1)jplus=0
       if(j<n-1)jplus=j+1

       do k=0,n-1
              if(k==0)kminus=n-1
              if(k>0)kminus=k-1
              if(k==n-1)kplus=0
              if(k<n-1)kplus=k+1

              do l=0,n-1
                 if(l==0)lminus=n-1
                 if(l>0)lminus=l-1
                 if(l==n-1)lplus=0
                 if(l<n-1)lplus=l+1

                 do c=1,3

f(c,j,k,l)=(v(c,jplus,k,l)+v(c,jminus,k,l)+v(c,j,kplus,l)+v(c,j,kminus,l)+v(c,j,
k,lplus)+ &
              & v(c,j,k,lminus)-6.0*v(c,j,k,l))/(h*h)

                 end do

              end do
```

```
      end do

end do

end subroutine sub1
!************************************************************************

subroutine plot(t,i,j,k,l,x,y,z,u)
implicit none
include "vect3dparam"
doubleprecision::u(1:3,0:n-1,0:n-1,0:n-1)
doubleprecision::x(0:n),y(0:n),z(0:n)
doubleprecision::sol,pi,t,diff,sumdiff
integer::c,i,j,k,l,jj,kk,ll

pi=4.0*atan(1.0)

call file(i)

do c=1,3

   do j=0,n
      if(j<n)jj=j
      if(j==n)jj=0

      do k=0,n
         if(k<n)kk=k
         if(k==n)kk=0

         do l=0,n
            if(l<n)ll=l
            if(l==n)ll=0

            call solution(c,jj,kk,ll,t,x,y,z,sol)

            if(j==n.and.k==n)then
               sumdiff=0.0
               write(unit=9,fmt="(5e15.7)")x(j),y(k),z(l),u(c,jj,kk,ll),sol
               diff=dabs(sol-u(c,jj,kk,ll))
               sumdiff=sumdiff+diff
            end if

         end do

      end do

   end do

write(unit=9,fmt=*)""

end do

sumdiff=sumdiff/(3.0*n)
print*,i,sumdiff

end subroutine plot
!************************************************************************

subroutine file(step)

implicit none
```



```
character*14::ch
integer::step

ch="vect3d****.dat"
write(ch(7:10),"(i4.4)")step
open(unit=9,file=ch)

end subroutine file
!************************************************************************

subroutine solution(c,jj,kk,ll,t,x,y,z,sol)

implicit none
include "vect3dparam"

doubleprecision::x(0:n),y(0:n),z(0:n)
doubleprecision::sol,pi,t
integer::c,jj,kk,ll

pi=4.0*atan(1.0)

sol=(0.5*(2.0-c)*(3.0-c)*(sin(2.0*pi*x(jj))+cos(2.0*pi*y(kk))+cos(2.0*pi*z(ll)))
*dexp(-4.0*pi*pi*t))+ &
                  & (3.0-c)*(c-1.0)*(cos(2.0*pi*y(kk)))*dexp(-4.0*pi*pi*t) + &
                  & 0.5*(c-2.0)*(c-1.0)*(cos(2.0*pi*z(ll)))*dexp(-4.0*pi*pi*t)

end subroutine solution
```