```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 1: Solve One-dimensional Heat Conduction Equation
! Method: FTCS
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=21,itp=5000
real,dimension(1:m)::told,tnew,x
real,parameter::L=1.0,alfa=117e-06,ts=300,tmax=500
real,parameter::dx=0.05,dt=0.01
real:: time,s
integer::i,it
open(1,file='result1.dat')
open(2,file='temperature1.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
! initial coditions
do i=1,m
told(i)=500.0
end do
! boundary conditions
told(1)=ts
told(m)=ts
time=0
it=0
s=alfa*dt/dx**2
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
tnew(i)=told(i)+s*(told(i+1)-2*told(i)+told(i-1))
end do
do i=2,m-1
told(i)=tnew(i)
end do
if(mod(it,itp).eq.0.0)then
write(1,10)it,time
write(1,20)(told(i),i=1,m,4)
end if
end do
do i=1,m
write(2,*)x(i),told(i)
end do
10 format('it=',i8,'time=',f7.3,'s')
20 format('temperature=',9(1x,f7.2))
stop
end
```

```
it=5000 time= 49.998s
temperature=  300.00  485.96  499.88  499.88  485.96  300.00
it=10000 time=100.003s
temperature=  300.00  461.48  497.83  497.83  461.48  300.00
it=15000 time=149.997s
temperature=  300.00  442.79  492.72  492.72  442.79  300.00
it=20000 time=199.969s
temperature=  300.00  428.92  485.56  485.56  428.92  300.00
it=25000 time=249.942s
temperature=  300.00  418.17  477.35  477.35  418.17  300.00
it=30000 time=299.981s
temperature=  300.00  409.43  468.72  468.72  409.43  300.00
it=35000 time=350.030s
temperature=  300.00  402.04  460.08  460.08  402.04  300.00
it=40000 time=400.079s
temperature=  300.00  395.56  451.61  451.61  395.56  300.00
it=45000 time=450.128s
temperature=  300.00  389.77  443.42  443.42  389.77  300.00
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 2: Solve One-dimensional Heat Conduction Equation
! Method: Dufort- Frankel
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=21,itp=5000
real,dimension(1:m)::told1,told2,tnew,x
real,parameter::L=1.0,alfa=117e-06,ts=300,tmax=500
real,parameter::dx=0.05,dt=0.01
real:: time,s
integer::i,it
open(1,file='result2.dat')
open(2,file='temperature2.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
! initial coditions
do i=1,m
told1(i)=500.0
told2(i)=500.0
end do
! boundary conditions
told2(1)=ts
told2(m)=ts
time=0
it=0
s=alfa*dt/dx**2
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
tnew(i)=((1-2*s)*told1(i)+2*s*(told2(i+1)+told2(i-1)))/(1+2*s)
end do
do i=2,m-1
told1(i)=told2(i)
told2(i)=tnew(i)
end do
if(mod(it,itp).eq.0.0)then
write(1,10)it,time
write(1,20)(told2(i),i=1,m,4)
end if
end do
do i=1,m
write(2,*)x(i),told2(i)
end do
10 format('it=',i8,'time=',f7.3,'s')
20 format('temperature=',9(1x,f7.2))
stop
end
```

```
it=5000 time= 49.998s
temperature=  300.00  485.96  499.88  499.88  485.96  300.00
it=10000 time=100.003s
temperature=  300.00  461.48  497.83  497.83  461.48  300.00
it=15000 time=149.997s
temperature=  300.00  442.79  492.73  492.73  442.79  300.00
it=20000 time=199.969s
temperature=  300.00  428.92  485.57  485.57  428.92  300.00
it=25000 time=249.942s
temperature=  300.00  418.16  477.36  477.36  418.16  300.00
it=30000 time=299.981s
temperature=  300.00  409.43  468.74  468.74  409.43  300.00
it=35000 time=350.030s
temperature=  300.00  402.02  460.09  460.09  402.02  300.00
it=40000 time=400.079s
temperature=  300.00  395.55  451.60  451.60  395.55  300.00
it=45000 time=450.128s
temperature=  300.00  389.74  443.41  443.41  389.74  300.00
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Program 3: Solve One-dimensional Heat Conduction Equation
! Method: Laasonen
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=21,itp=5000
real,dimension(1:m)::told,tnew,x,a,b,c,d
real,parameter::L=1.0,alfa=117e-06,ts=300,tmax=500
real,parameter::dx=0.05,dt=0.01
real::time,s
integer::i,it
open(1,file='result3.dat')
open(2,file='temperature3.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
! initial coditions
do i=1,m
told(i)=500.0
end do
! boundary conditions
told(1)=ts
told(m)=ts
s=alfa*dt/dx**2
do i=2,m-1
a(i)=s
b(i)=-(1+2*s)
c(i)=s
end do
time=0
it=0
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
d(i)=-told(i)
end do
d(2)=d(2)-a(2)*told(1)
d(m-1)=d(m-1)-c(m-1)*told(m)
call TDMA(m,a,b,c,d,tnew)
do i=2,m-1
told(i)=tnew(i)
end do
if(mod(it,itp).eq.0.0)then
print*,it,time
write(1,10)it,time
write(1,20)(told(i),i=1,m,4)
```

```
end if
end do
do i=1,m
write(2,*)x(i),told(i)
end do
10  format('it=',i8,'time=',f7.3,'s')
20  format('temperature=',9(1x,f7.2))
stop
end
!  Tri-diagonal matrix solver TDMA
subroutine TDMA(m,a,b,c,d,tnew)
implicit none
integer::i,m
real,dimension(1:m)::tnew,a,b,c,d,b1,d1
! forward elimination
b1(2)=b(2)
d1(2)=d(2)
do i=3,m-1
b1(i)=b(i)-(a(i)/b(i-1))*c(i-1)
d1(i)=d(i)-(a(i)/b(i-1))*d(i-1)
end do
! backward substitution
tnew(m-1)=d1(m-1)/b1(m-1)
do i=m-2,2,-1
tnew(i)=(d1(i)-c(i)*tnew(i+1))/b1(i)
end do
return
end
```

```
it=5000 time= 49.998s
temperature=  300.00  485.50  499.31  499.31  485.40  300.00
it=10000 time=100.003s
temperature=  300.00  460.68  496.72  496.69  460.44  300.00
it=15000 time=149.997s
temperature=  300.00  441.68  491.10  491.04  441.35  300.00
it=20000 time=199.969s
temperature=  300.00  427.52  483.48  483.36  427.12  300.00
it=25000 time=249.942s
temperature=  300.00  416.51  474.83  474.68  416.05  300.00
it=30000 time=299.981s
temperature=  300.00  407.53  465.81  465.62  407.03  300.00
it=35000 time=350.030s
temperature=  300.00  399.90  456.77  456.56  399.37  300.00
it=40000 time=400.079s
temperature=  300.00  393.22  447.93  447.71  392.67  300.00
it=45000 time=450.128s
temperature=  300.00  387.21  439.41  439.17  386.65  300.00
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 4: Solve Two-dimensional Heat Conduction Equation
! Method: FTCS
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=31,n=41,itp=5000
real,dimension(1:m,1:n)::told,tnew,x,y
real,parameter::L=0.3,W=0.4,alfa=11.2347e-05,tmax=500
real,parameter::to=0.0,t1=40.0,t2=0.0,t3=10.0,t4=0.0,dt=0.02
real:: time,s1,s2,dx,dy
integer::i,j,it
open(1,file='result4.dat')
open(2,file='temperature4.dat')
dx=L/float(m-1)
dy=W/float(n-1)
do i=1,m
do j=1,n
x(i,j)=dx*float(i-1)
y(i,j)=dy*float(j-1)
end do
end do
s1=alfa*dt/dx**2
s2=alfa*dt/dy**2
! initial coditions
do i=1,m
do j=1,n
told(i,j)=to
end do
end do
! boundary conditions
do i=1,m
told(i,1)=t1
told(i,n)=t3
end do
do j=1,n
told(1,j)=t2
told(m,j)=t4
end do
time=0
it=0
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
do j=2,n-1
tnew(i,j)=s1*(told(i+1,j)+told(i-1,j))+(1-2.0*s1-2.0*s2)      &
         *told(i,j)+s2*(told(i,j+1)+told(i,j-1))
end do
end do
```

```
      do i=2,m-1
      do j=2,n-1
      told(i,j)=tnew(i,j)
      end do
      end do
      if(mod(it,itp).eq.0.0)then
      write(1,10)it,time
      write(1,20)((told(i,j),i=1,m,5),j=n,1,-5)
      end if
      end do
      write(2,'(A)')'VARIABLE="x","y"'
      write(2,30)n,m
      do i=1,m
      do j=1,n
      write(2,*)x(i,j),y(i,j),told(i,j)
      end do
      end do
   10 format('it=',i8,'time=',f11.4,'s')
   20 format(7(1x,f5.1))
   30 format('ZONE i= ',i3,'  j=',I3,'  f=point')
      stop
      end
```

```
it=5000 time=     99.9952s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.8   6.5   6.9   6.5   4.8    .0
    .0   2.7   4.4   4.9   4.4   2.7    .0
    .0   2.2   3.8   4.3   3.8   2.2    .0
    .0   2.8   4.9   5.6   4.9   2.8    .0
    .0   4.8   8.1   9.2   8.1   4.8    .0
    .0   9.1  14.4  16.0  14.4   9.1    .0
    .0  18.5  25.0  26.5  25.0  18.5    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0

it=10000 time=    200.0059s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   5.1   7.0   7.5   7.0   5.1    .0
    .0   3.3   5.3   6.0   5.3   3.3    .0
    .0   3.0   5.1   5.9   5.1   3.0    .0
    .0   3.8   6.4   7.4   6.4   3.8    .0
    .0   5.7   9.6  11.0   9.6   5.7    .0
    .0   9.8  15.6  17.4  15.6   9.8    .0
    .0  18.9  25.7  27.3  25.7  18.9    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0

it=15000 time=    299.9937s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   5.1   7.1   7.6   7.1   5.1    .0
    .0   3.4   5.5   6.2   5.5   3.4    .0
    .0   3.1   5.3   6.1   5.3   3.1    .0
    .0   3.9   6.7   7.6   6.7   3.9    .0
    .0   5.9   9.9  11.2   9.9   5.9    .0
    .0   9.9  15.8  17.6  15.8   9.9    .0
    .0  18.9  25.8  27.4  25.8  18.9    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0

it=20000 time=    399.9388s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   5.1   7.1   7.6   7.1   5.1    .0
    .0   3.4   5.5   6.2   5.5   3.4    .0
    .0   3.1   5.4   6.1   5.4   3.1    .0
    .0   3.9   6.7   7.7   6.7   3.9    .0
    .0   5.9   9.9  11.3   9.9   5.9    .0
    .0   9.9  15.8  17.6  15.8   9.9    .0
    .0  18.9  25.8  27.4  25.8  18.9    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0

it=25000 time=    499.8838s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   5.1   7.1   7.6   7.1   5.1    .0
    .0   3.4   5.5   6.2   5.5   3.4    .0
    .0   3.1   5.4   6.1   5.4   3.1    .0
    .0   3.9   6.7   7.7   6.7   3.9    .0
    .0   5.9   9.9  11.3   9.9   5.9    .0
    .0   9.9  15.8  17.6  15.8   9.9    .0
    .0  18.9  25.8  27.4  25.8  18.9    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 5: Solve Two-dimensional Heat Conduction Equation
! Method: ADI
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=31,n=41,itp=5000
real,dimension(1:m,1:n)::t,x,y
real,dimension(1:m)::tx,a1,b1,c1,d1
real,dimension(1:n)::ty,a2,b2,c2,d2
real,parameter::L=0.3,W=0.4,alfa=11.2347e-05,tmax=500
real,parameter::to=0.0,t1=40.0,t2=0.0,t3=10.0,t4=0.0,dt=0.02
real:: time,s1,s2,dx,dy
integer::i,j,it
open(1,file='result5.dat')
open(2,file='temperature5.dat')
dx=L/float(m-1)
dy=W/float(n-1)
do i=1,m
do j=1,n
x(i,j)=dx*float(i-1)
y(i,j)=dy*float(j-1)
end do
end do
s1=(alfa*dt/dx**2)/2.0
s2=(alfa*dt/dy**2)/2.0
do i=2,m-1
a1(i)=-s1
b1(i)=1+2.0*s1
c1(i)=-s1
end do
do j=2,n-1
a2(j)=-s2
b2(j)=1+2.0*s2
c2(j)=-s2
end do
! initial coditions
do i=1,m
do j=1,n
t(i,j)=to
end do
end do
! boundary conditions
do j=1,n
t(1,j)=t2
t(m,j)=t4
end do
do i=1,m
t(i,1)=t1
t(i,n)=t3
end do
```

```
      time=0.0
      it=0
      print*,'iteration start'
      do while(time<tmax)
      time=time+dt
      it=it+1
      ! x-sweep
      do j=2,n-1
      do i=2,m-1
      d1(i)=s2*t(i,j+1)+(1.0-2.0*s2)*t(i,j)+s2*t(i,j-1)
      end do
      d1(2)=d1(2)-a1(2)*t(1,j)
      d1(m-1)=d1(m-1)-c1(m-1)*t(m,j)
      call TDMA(m,a1,b1,c1,d1,tx)
      do i=2,m-1
      t(i,j)=tx(i)
      end do
      end do
      ! y-sweep
      do i=2,m-1
      do j=2,n-1
      d2(j)=s1*t(i+1,j)+(1.0-2.0*s1)*t(i,j)+s1*t(i-1,j)
      end do
      d2(2)=d2(2)-a2(2)*t(i,1)
      d2(n-1)=d2(n-1)-c2(n-1)*t(i,n)
      call TDMA(n,a2,b2,c2,d2,ty)
      do j=2,n-1
      t(i,j)=ty(j)
      end do
      end do
      if(mod(it,itp).eq.0.0)then
      write(1,10)it,time
      write(1,20)((t(i,j),i=1,m,5),j=n,1,-5)
      end if
      end do
      write(2,'(A)')'VARIABLE="x","y"'
      write(2,30)n,m
      do i=1,m
      do j=1,n
      write(2,*)x(i,j),y(i,j),t(i,j)
      end do
      end do
   10 format('it=',i8,'time=',f11.4,'s')
   20 format(7(1x,f5.1))
   30 format('ZONE i= ',i3,'  j=',I3,'  f=point')
      stop
      end
      !  Tri-diagonal matrix solver TDMA
      subroutine TDMA(m,a,b,c,d,tnew)
      implicit none
      integer::i,m
      real,dimension(1:m)::tnew,a,b,c,d,b1,d1
      ! forward elimination
      b1(2)=b(2)
```

```
d1(2)=d(2)
do i=3,m-1
b1(i)=b(i)-(a(i)/b(i-1))*c(i-1)
d1(i)=d(i)-(a(i)/b(i-1))*d(i-1)
end do
! backward substitution
tnew(m-1)=d1(m-1)/b1(m-1)
do i=m-2,2,-1
tnew(i)=(d1(i)-c(i)*tnew(i+1))/b1(i)
end do
return
end
```

```
it=5000 time=     99.9952s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.1   5.4   5.6   5.3   4.0    .0
    .0   1.9   2.9   3.2   2.8   1.8    .0
    .0   1.2   2.0   2.3   2.0   1.1    .0
    .0   1.5   2.4   2.7   2.3   1.4    .0
    .0   2.8   4.5   5.0   4.4   2.6    .0
    .0   6.3   9.6  10.4   9.4   6.0    .0
    .0  15.6  20.3  21.3  20.1  15.2    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
it=10000 time=    200.0059s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.2   5.5   5.7   5.4   4.0    .0
    .0   2.0   3.1   3.4   3.0   1.9    .0
    .0   1.4   2.3   2.5   2.2   1.3    .0
    .0   1.6   2.7   3.1   2.6   1.5    .0
    .0   3.0   4.8   5.4   4.7   2.8    .0
    .0   6.4   9.8  10.7   9.6   6.1    .0
    .0  15.6  20.5  21.4  20.3  15.3    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
it=15000 time=    299.9937s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.2   5.5   5.7   5.4   4.0    .0
    .0   2.0   3.1   3.4   3.0   1.9    .0
    .0   1.4   2.3   2.6   2.2   1.3    .0
    .0   1.7   2.8   3.1   2.6   1.5    .0
    .0   3.0   4.9   5.4   4.7   2.8    .0
    .0   6.4   9.9  10.7   9.6   6.1    .0
    .0  15.7  20.5  21.4  20.3  15.3    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
it=20000 time=    399.9388s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.2   5.5   5.7   5.4   4.0    .0
    .0   2.0   3.1   3.4   3.0   1.9    .0
    .0   1.4   2.3   2.6   2.2   1.3    .0
    .0   1.7   2.8   3.1   2.6   1.5    .0
    .0   3.0   4.9   5.4   4.7   2.8    .0
    .0   6.4   9.9  10.7   9.6   6.1    .0
    .0  15.7  20.5  21.4  20.3  15.3    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
it=25000 time=    499.8838s
  10.0  10.0  10.0  10.0  10.0  10.0  10.0
    .0   4.2   5.5   5.7   5.4   4.0    .0
    .0   2.0   3.1   3.4   3.0   1.9    .0
    .0   1.4   2.3   2.6   2.2   1.3    .0
    .0   1.7   2.8   3.1   2.6   1.5    .0
    .0   3.0   4.9   5.4   4.7   2.8    .0
    .0   6.4   9.9  10.7   9.6   6.1    .0
    .0  15.7  20.5  21.4  20.3  15.3    .0
  40.0  40.0  40.0  40.0  40.0  40.0  40.0
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 6: Solve Two-dimensional Laplace Equation
! Method:PSOR
! CFD Course (MSc),2015
! Dr.Mohammed A.Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=31,n=61,itp=50,itmax=10000
real,dimension(1:m,1:n)::t,x,y
real,parameter::L=0.25,H=0.5,errormax=0.001,w=1.3
real,parameter::to=0.0,t1=200.0,t2=0.0,t3=0.0
real,parameter::qw=10000,ktc=13.4
real::dx,dy,told,error,beta
integer::i,j,it
open(1,file='result6.dat')
open(2,file='contour6.dat')
dx=L/float(m-1)
dy=H/float(n-1)
beta=dx/dy
do i=1,m
do j=1,n
x(i,j)=dx*float(i-1)
y(i,j)=dy*float(j-1)
end do
end do
!  initial coditions
do i=1,m
do j=1,n
t(i,j)=to
end do
end do
!  boundary conditions
do j=1,n
t(1,j)=t3
t(m,j)=t2
end do
do i=1,m
t(i,1)=t1
end do
it=0
print*,'iteration start using PSOR'
10  it=it+1
error=0.0
do i=1,m
t(i,n)=qw*dy/ktc+t(i,n-1)
end do
do i=2,m-1
do j=2,n-1
told=t(i,j)
t(i,j)=(1.0-w)*t(i,j)+w/(2.0*(1.0+beta*beta))*((t(i+1,j)+t(i-1,j))&
      +beta*beta*(t(i,j+1)+t(i,j-1)))
```

```
      error=error+abs((t(i,j)-told)/t(i,j))
      end do
      end do
      if(mod(it,itp).eq.0.0)then
      write(*,*)it,error
      write(1,20)it,error
      end if
      if(it.eq.itmax)then
      write(*,*) ' warning solution is not converged
      after',itmax,'iterations'
      else if (error>errormax)then
      goto 10
      else
      write(*,*) 'solution is converged after',it,'iterations'
      end if
      write(*,*)it,error
      write(1,20)it,error
      write(2,'(A)')'VARIABLE="x","y"'
      write(2,40)n,m
      do i=1,m
      do j=1,n
      write(2,*)x(i,j),y(i,j),t(i,j)
      end do
      end do
      write(1,30)((t(i,j),i=1,m,5),j=n,1,-5)
20    format('it=',i7,'error=',f12.7)
30    format(7(1x,f6.2))
40    format('ZONE i= ',i3,'  j=',I3,'  f=point')
      stop
      end
```

```
it=50  error=  64.0003300
it=100 error=  16.9869100
it=150 error=   7.0151810
it=200 error=   3.4168470
it=250 error=   1.8015730
it=300 error=    .9912030
it=350 error=    .5587413
it=400 error=    .3195193
it=450 error=    .1843098
it=500 error=    .1068994
it=550 error=    .0622105
it=600 error=    .0362890
it=650 error=    .0211901
it=700 error=    .0123927
it=750 error=    .0072484
it=800 error=    .0042389
it=850 error=    .0024859
it=900 error=    .0014444
it=936 error=    .0009914
```

6.22	49.86	67.97	73.41	67.97	49.86	6.22
.00	26.32	41.64	46.51	41.64	26.32	.00
.00	15.20	25.54	29.10	25.54	15.20	.00
.00	9.58	16.44	18.89	16.44	9.58	.00
.00	6.90	11.92	13.74	11.92	6.90	.00
.00	6.20	10.73	12.39	10.73	6.20	.00
.00	7.25	12.55	14.48	12.55	7.25	.00
.00	10.36	17.88	20.62	17.88	10.36	.00
.00	16.46	28.22	32.43	28.22	16.46	.00
.00	27.62	46.45	52.89	46.45	27.62	.00
.00	48.73	77.49	86.28	77.49	48.73	.00
.00	94.27	128.23	136.27	128.23	94.27	.00
200.00	200.00	200.00	200.00	200.00	200.00	200.00

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 7: Solve Two-dimensional Laplace Equation
! Method: LSOR
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=31,n=61,itp=10,itmax=1000
real,dimension(1:m,1:n)::t,x,y
real,dimension(1:m)::tx,a,b,c,d
real,parameter::L=0.25,H=0.5,errormax=0.001,w=1.3
real,parameter::to=0.0,t1=200.0,t2=0.0,t3=0.0
real,parameter::qw=10000,ktc=13.4
real::dx,dy,error,beta
integer::i,j,it
open(1,file='result7.dat')
open(2,file='contour7.dat')
dx=L/float(m-1)
dy=H/float(n-1)
beta=dx/dy
do i=1,m
do j=1,n
x(i,j)=dx*float(i-1)
y(i,j)=dy*float(j-1)
end do
end do
! initial coditions
do i=1,m
do j=1,n
t(i,j)=to
end do
end do
! boundary conditions
do j=1,n
t(1,j)=t3
t(m,j)=t2
end do
do i=1,m
t(i,1)=t1
end do
do i=2,m-1
a(i)=w
b(i)=-2.0*(1.0+beta*beta)
c(i)=w
end do
it=0
print*,'iteration start using LSOR'
10 it=it+1
error=0.0
do i=1,m
t(i,n)=qw*dy/ktc+t(i,n-1)
end do
```

```
      do j=2,n-1
      do i=2,m-1
      d(i)=-1.0*(1.0-w)*(2.0*(1+beta*beta))*t(i,j)-
      w*beta*beta*(t(i,j+1)+t(i,j-1))
      end   do
      d(2)=d(2)-a(2)*t(1,j)
      d(m-1)=d(m-1)-c(m-1)*t(m,j)
      call TDMA(m,a,b,c,d,tx)
      do i=2,m-1
      error=error+abs((t(i,j)-tx(i))/tx(i))
      t(i,j)=tx(i)
      end do
      end do
      if(mod(it,itp).eq.0.0)then
      write(*,*)it,error
      write(1,20)it,error
      end if
      if(it.eq.itmax)then
      write(*,*) ' warning solution is not converged
      after',itmax,'iterations'
      else if (error>errormax)then
      goto 10
      else
      write(*,*) 'solution is converged after',it,'iterations'
      end if
      write(*,*)it,error
      write(1,20)it,error
      write(2,'(A)')'VARIABLE="x","y"'
      write(2,40)n,m
      do i=1,m
      do j=1,n
      write(2,*)x(i,j),y(i,j),t(i,j)
      end do
      end do
      write(1,30)((t(i,j),i=1,m,5),j=n,1,-5)
   20 format('it=',i7,'error=',f12.7)
   30 format(7(1x,f6.2))
   40 format('ZONE i= ',i3,'  j=',I3,'  f=point')
      stop
      end

      !  Tri-diagonal matrix solver TDMA
      subroutine TDMA(m,a,b,c,d,tt)
      implicit none
      integer::i,m
      real,dimension(1:m)::tt,a,b,c,d,b1,d1
      ! forward elimination
      b1(2)=b(2)
      d1(2)=d(2)/b1(2)
      do i=3,m-1
      b1(i)=b(i)-a(i)*c(i-1)/b1(i-1)
      d1(i)=(d(i)-a(i)*d1(i-1))/b1(i)
      end do
      ! backward substitution
```

```
      tt(m-1)=d1(m-1)
      do i=m-2,2,-1
      tt(i)=d1(i)-c(i)*tt(i+1)/b1(i)
      end do
      return
      end
```

```
it=10  error=85.6545600
it=20  error=43.1687200
it=30  error=17.6489400
it=40  error=4.9271480
it=50  error=1.1447410
it=60  error=.2357099
it=70  error=.0864692
it=80  error=.0198260
it=90  error=.0047677
it=100 error=.0012749
it=102 error=.0009735

   6.22  49.86  67.97  73.41  67.97  49.86   6.22
    .00  26.32  41.64  46.51  41.64  26.32    .00
    .00  15.20  25.55  29.11  25.55  15.20    .00
    .00   9.58  16.44  18.89  16.44   9.58    .00
    .00   6.90  11.92  13.74  11.92   6.90    .00
    .00   6.20  10.73  12.39  10.73   6.20    .00
    .00   7.25  12.55  14.49  12.55   7.25    .00
    .00  10.36  17.88  20.62  17.88  10.36    .00
    .00  16.46  28.22  32.43  28.22  16.46    .00
    .00  27.62  46.45  52.89  46.45  27.62    .00
    .00  48.73  77.49  86.28  77.49  48.73    .00
    .00  94.27 128.23 136.27 128.23  94.27    .00
 200.00 200.00 200.00 200.00 200.00 200.00 200.00
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 8: Solve Two-dimensional Laplace Equation
! Method: ADI
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=31,n=61,itp=10,itmax=10000
real,dimension(1:m,1:n)::t,x,y
real,dimension(1:m)::tx,a1,b1,c1,d1
real,dimension(1:n)::ty,a2,b2,c2,d2
real,parameter::L=0.25,H=0.5,errormax=0.001,w=1.3
real,parameter::to=0.0,t1=200.0,t2=0.0,t3=0.0
real,parameter::qw=10000,ktc=13.4
real::dx,dy,error,beta
integer::i,j,it
open(1,file='result8.dat')
open(2,file='contour8.dat')
dx=L/float(m-1)
dy=H/float(n-1)
beta=dx/dy
do i=1,m
do j=1,n
x(i,j)=dx*float(i-1)
y(i,j)=dy*float(j-1)
end do
end do
!  initial coditions
do i=1,m
do j=1,n
t(i,j)=to
end do
end do
!  boundary conditions
do j=1,n
t(1,j)=t3
t(m,j)=t2
end do
do i=1,m
t(i,1)=t1
end do
do i=2,m-1
a1(i)=w
b1(i)=-2.0*(1.0+beta*beta)
c1(i)=w
end do
do j=2,n-1
a2(j)=w*beta*beta
b2(j)=-2.0*(1.0+beta*beta)
c2(j)=w*beta*beta
end do
it=0
```

```
      print*,'iteration start using ADI'
10    it=it+1
      error=0.0
      do i=1,m
      t(i,n)=qw*dy/ktc+t(i,n-1)
      end do
!     x-sweep
      do j=2,n-1
      do i=2,m-1
      d1(i)=-1.0*(1.0-w)*(2.0*(1+beta*beta))*t(i,j)-
      w*beta*beta*(t(i,j+1)+t(i,j-1))
      end   do
      d1(2)=d1(2)-a1(2)*t(1,j)
      d1(m-1)=d1(m-1)-c1(m-1)*t(m,j)
      call TDMA(m,a1,b1,c1,d1,tx)
      do i=2,m-1
      t(i,j)=tx(i)
      end do
      end do
!     y-sweep
      do i=2,m-1
      do j=2,n-1
      d2(j)=-1.0*(1.0-w)*(2.0*(1+beta*beta))*t(i,j)-w*(t(i+1,j)+t(i-1,j))
      end   do
      d2(2)=d2(2)-a2(2)*t(i,1)
      d2(n-1)=d2(n-1)-c2(n-1)*t(i,n)
      call TDMA(n,a2,b2,c2,d2,ty)
      do j=2,n-1
      error=error+abs((t(i,j)-ty(j))/ty(j))
      t(i,j)=ty(j)
      end do
      end do
      if(mod(it,itp).eq.0.0)then
      write(*,*)it,error
      write(1,20)it,error
      end if
      if(it.eq.itmax)then
      write(*,*) ' warning solution is not converged
      after',itmax,'iterations'
      else if (error>errormax)then
      goto 10
      else
      write(*,*) 'solution is converged after',it,'iterations'
      end if
      write(*,*)it,error
      write(1,20)it,error
      write(2,'(A)')'VARIABLE="x","y"'
      write(2,40)n,m
      do i=1,m
      do j=1,n
      write(2,*)x(i,j),y(i,j),t(i,j)
      end do
      end do
      write(1,30)((t(i,j),i=1,m,5),j=n,1,-5)
```

```
20   format('it=',i7,'error=',f12.7)
30   format(7(1x,f6.2))
40   format('ZONE i= ',i3,'  j=',I3,'  f=point')
     stop
     end

!    Tri-diagonal matrix solver TDMA
     subroutine TDMA(m,a,b,c,d,tt)
     implicit none
     integer::i,m
     real,dimension(1:m)::tt,a,b,c,d,b1,d1
!    forward elimination
     b1(2)=b(2)
     d1(2)=d(2)/b1(2)
     do i=3,m-1
     b1(i)=b(i)-a(i)*c(i-1)/b1(i-1)
     d1(i)=(d(i)-a(i)*d1(i-1))/b1(i)
     end do
!    backward substitution
     tt(m-1)=d1(m-1)
     do i=m-2,2,-1
     tt(i)=d1(i)-c(i)*tt(i+1)/b1(i)
     end do
     return
     end
```

```
it=10  error=40.0133100
it=20  error=13.4371400
it=30  error=5.3622990
it=40  error=2.2564370
it=50  error=.9739987
it=60  error=.4252243
it=70  error=.1865446
it=80  error=.0819835
it=90  error=.0360568
it=100 error=.0159392
it=110 error=.0070485
it=120 error=.0031723
it=130 error=.0015364
it=136 error=.0009244

   6.22  49.86  67.97  73.41  67.97  49.86   6.22
    .00  26.32  41.64  46.51  41.64  26.32    .00
    .00  15.20  25.55  29.11  25.55  15.20    .00
    .00   9.58  16.44  18.89  16.44   9.58    .00
    .00   6.90  11.92  13.74  11.92   6.90    .00
    .00   6.20  10.73  12.39  10.73   6.20    .00
    .00   7.25  12.55  14.49  12.55   7.26    .00
    .00  10.36  17.88  20.62  17.88  10.36    .00
    .00  16.46  28.22  32.43  28.22  16.46    .00
    .00  27.62  46.45  52.89  46.45  27.62    .00
    .00  48.73  77.49  86.28  77.49  48.73    .00
    .00  94.27 128.23 136.27 128.23  94.27    .00
 200.00 200.00 200.00 200.00 200.00 200.00 200.00
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 9: Solve First-Order Hyperbolic Equation
! Method: First upwind, explicit
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=71,itp=20
real,dimension(1:m)::uold,unew,x
real,parameter::dx=1.0,dt=0.005,tmax=0.35,a=200
real::time,s
integer::i,it
open(1,file='result.dat')
open(2,file='velocity profile.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
! initial coditions
do i=1,m
if(x(i).ge.5.and.x(i).le.15.0)then
uold(i)=20.0/float(16-6)*float(i-6)
else if(x(i).ge.15.and.x(i).le.25.0)then
uold(i)=20.-20.0/float(26-16)*float(i-16)
else
uold(i)=0.0
end if
end do
s=a*dt/dx
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
unew(i)=uold(i)-s*(uold(i)-uold(i-1))
end do
do i=2,m-1
uold(i)=unew(i)
end do
if(mod(it,itp).eq.0.0)then
write(1,10)it,time
write(1,20)(uold(i),i=1,m)
end if
end do
do i=1,m
write(2,*)x(i),uold(i)
end do
10 format('it=',i4,'time=',f8.4,'s')
20 format(10(1x,f5.1))
stop
end
```

```
it=20 time=0.1000s
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0   2.0   4.0   6.0   8.0
  10.0  12.0  14.0  16.0  18.0  20.0  18.0  16.0  14.0  12.0
  10.0   8.0   6.0   4.0   2.0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0

it=40 time=0.2000s
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0   2.0   4.0   6.0   8.0
  10.0  12.0  14.0  16.0  18.0  20.0  18.0  16.0  14.0  12.0
  10.0   8.0   6.0   4.0   2.0    .0    .0    .0    .0    .0
    .0

it=60 time=0.3000s
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0   2.0   4.0   6.0   8.0
    .0
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 10: Solve First-Order Hyperbolic Equation
! Method: Euler's FTCS Implicit Method
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=71,itp=20
real,dimension(1:m)::uold,unew,x,a,b,c,d
real,parameter::dx=1,dt=0.005,tmax=0.35,a1=200
real::time,s
integer::i,it
open(1,file='result.dat')
open(2,file='velocity.dat')
open(3,file='velocity profile.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
!  initial coditions
do i=1,m
if(x(i).ge.5.and.x(i).le.15.0)then
uold(i)=20.0/float(16-6)*float(i-6)
else if(x(i).ge.15.and.x(i).le.25.0)then
uold(i)=20.-20.0/float(26-16)*float(i-16)
else
uold(i)=0
end if
end do
s=a1*dt/dx
do i=2,m-1
a(i)=0.5*s
b(i)=-1.0
c(i)=-0.5*s
end do
print*,'iteration start'
do while(time<tmax)
time=time+dt
it=it+1
do i=2,m-1
d(i)=-uold(i)
end do
d(2)=d(2)-a(2)*uold(1)
d(m-1)=d(m-1)-c(m-1)*uold(m)
call TDMA(m,a,b,c,d,unew)
do i=2,m-1
uold(i)=unew(i)
end do
if(mod(it,itp).eq.0.0)then
write(1,10)it,time
write(1,20)(uold(i),i=1,m)
end if
end do
```

```
it=20 time=0.100s
    .0    .1    .1    .1    .1    .0    .1    .0    .1    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .1
    .3    .6   1.1   1.7   2.5   3.5   4.7   6.1   7.5   8.9
  10.2  11.3  12.2  12.8  13.1  13.0  12.6  11.9  11.0   9.9
   8.7   7.5   6.3   5.2   4.2   3.3   2.6   1.9   1.5   1.1
    .8    .5    .4    .3    .2    .1    .1    .1    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0

it=40 time=0.200s
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .1    .2    .4    .6    .9
   1.3   1.8   2.4   3.1   3.9   4.8   5.7   6.6   7.6   8.4
   9.2   9.8  10.2  10.5  10.6  10.6  10.2   9.9   9.2   8.7
   7.8   7.2   6.1   5.8   4.5   4.5   2.9   3.6   1.4   3.1
    .0

it=60 time=0.300s
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0    .0    .0    .0    .0    .0    .0
    .0    .0    .0    .0   -.1    .1   -.1    .1   -.1    .2
   -.2    .3   -.4    .5   -.6    .8   -.9   1.2  -1.4   1.8
  -2.0   2.5  -2.7   3.6  -3.5   4.9  -4.3   6.5  -4.9   8.3
  -5.2  10.4  -5.0  12.4  -4.4  14.3  -3.2  15.7  -1.7  16.4
    .0
```

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Programm 11: Solve Inviscid Burgers Equation
! Method: MacCormack
! CFD Course (MSc),2015
! Dr.Mohammed A. Ahmed
! Mechanical Engineering Department
! College of Engineering, University of Anbar
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
implicit none
integer,parameter::m=40,itp=5
real,dimension(1:m)::uold,unew,ustar,E,x
real,parameter::dx=1,dt=0.1,tmax=2.4
real,parameter::umax=5.0,umin=0.0
real::time,s
integer::i,it
open(1,file='result.dat')
open(2,file='velocity profile.dat')
do i=1,m
x(i)=dx*float(i-1)
end do
! initial coditions
do i=1,m
if(x(i).le.20)then
uold(i)=umax
else
uold(i)=umin
end if
end do
s=umax*dt/dx
print*,'iteration start'
do while(time<=tmax)
time=time+dt
it=it+1
do i=1,m
E(i)=(uold(i)*uold(i))/2.0
end do
do i=2,m-1
ustar(i)=uold(i)-(dt/dx)*(E(i+1)-E(i))
end do
do i=2,m-1
E(i)=(ustar(i)*ustar(i))/2.0
end do
do i=2,m-1
unew(i)=0.5*(uold(i)+ustar(i)-(dt/dx)*(E(i)-E(i-1)))
end do
do i=2,m-1
uold(i)=unew(i)
end do
if(mod(it,itp).eq.0.0)then
write(1,10)it,time
write(1,20)(uold(i),i=1,m)
end if
end do
do i=1,m
```

```
      write(2,*)x(i),uold(i)
      end do
 10 format('it=',i5,'time=',f8.4,'s')
 20 format(10(1x,f6.1))
      stop
      end
```

```
it=5 time=0.5000s
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    4.8
    5.4    4.9    1.1     .0     .0     .0     .0     .0     .0     .0
     .0     .0     .0     .0     .0     .0     .0     .0     .0     .0

it=10 time=1.0000s
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    4.8    5.3    5.4    2.0     .0     .0     .0     .0     .0     .0
     .0     .0     .0     .0     .0     .0     .0     .0     .0     .0

it=15 time=1.5000s
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    4.8    5.1    5.6    3.1     .2     .0     .0     .0     .0
     .0     .0     .0     .0     .0     .0     .0     .0     .0     .0

it=20 time=2.0000s
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.1    4.8    4.9    5.6    4.1     .5     .0     .0     .0
     .0     .0     .0     .0     .0     .0     .0     .0     .0     .0

it=25 time=2.5000s
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0    5.0
    5.0    5.0    5.1    4.9    4.7    5.4    4.9    1.1     .0     .0
     .0     .0     .0     .0     .0     .0     .0     .0     .0     .0
```

```
      do i=1,m
      write(2,*)x(i),uold(i)
      end do
 10 format('it=',i5,'time=',f8.3,'s')
 20 format(10(1x,f5.1))
      stop
      end

 ! Tri-diagonal matrix solver TDMA
      subroutine TDMA(m,a,b,c,d,tt)
      implicit none
      integer::i,m
      real,dimension(1:m)::tt,a,b,c,d,b1,d1
 ! forward elimination
      b1(2)=b(2)
      d1(2)=d(2)/b1(2)
      do i=3,m-1
      b1(i)=b(i)-a(i)*c(i-1)/b1(i-1)
      d1(i)=(d(i)-a(i)*d1(i-1))/b1(i)
      end do
 ! backward substitution
      tt(m-1)=d1(m-1)
      do i=m-2,2,-1
      tt(i)=d1(i)-c(i)*tt(i+1)/b1(i)
      end do
      return
      end
```