

PS6_1

1.1 Read matrix M and N:

```
Program Main

implicit none

integer :: u, i
real :: a(5,3), b(3,5), c(5,5)

! File unit
u = 10
! Open the file
open(unit=u, file='M.dat')
! Read data line by line
do i = 1,5
    read(u, *) a(i,1:3)
enddo
! Close the file
close(u)
! Display the values
do i=1,5
    write(*,*) a(i,1:3)
enddo

! File unit
u = 20
! Open the file
open(unit=u, file='N.dat')
! Read data line by line
do i = 1,3
    read(u, *) b(i,1:5)
enddo
! Close the file
close(u)
! Display the values
do i=1,3
    write(*,*) b(i,1:5)
enddo
```

Output of M and N:

```
[ese-wangy1@login03 fortran_demo1]$ ./Main.x
 19.4799995    15.7900000    19.2800007
 19.2800007    12.9200001    15.8599997
 15.8599997    11.2900000    14.0400000
 11.9300003    18.6000004    18.2299995
 19.2800007    12.9200001    15.8599997
 7.71999979    4.11000013    1.44000006    4.80000019    5.55000019
 5.55000019    4.80000019    4.03999996    0.589999974    8.57999992
 0.589999974    8.57999992    2.25999999    7.71999979    4.11000013
```

1.2 Write a subroutine Matrix_Multip.f90:

```

GNU nano 2.3.1                                     File: Matrix_multip.f90
Subroutine Matrix_multip(a,b,c)
implicit none
integer :: i
real    :: a(5,3),b(3,5),c(5,5)
c=matmul(a,b)

end

[ese-wangy1@login03 fortran_demo1]$ gfortran -c Matrix_multip.f90

```

1.3 Call the subroutine Matrix_Multip.f90 from Main.f90:

```

Program Main

implicit none

integer                :: u, i
real :: a(5,3), b(3,5),c(5,5)

! File unit
u = 10
! Open the file
open(unit=u, file='M.dat')
! Read data line by line
do i = 1,5
  read(u, *) a(i,1:3)
enddo
! Close the file
close(u)
! Display the values
do i=1,5
  write(*,*) a(i,1:3)
enddo

! File unit
u = 20
! Open the file
open(unit=u, file='N.dat')
! Read data line by line
do i = 1,3
  read(u, *) b(i,1:5)
enddo
! Close the file
close(u)
! Display the values
do i=1,3
  write(*,*) b(i,1:5)
enddo

call Matrix_multip(a,b,c)
open(unit=u, file='MN.dat', status='replace')
do i=1,5
  write(u, "(f9.2,f9.2,f9.2,f9.2,f9.2)") c(i,1:5)
enddo
close(u)
End Program Main

```

```

[ese-wangy1@login03 fortran_demo1]$ nano Main.f90
[ese-wangy1@login03 fortran_demo1]$ gfortran Main.f90 Matrix_multip.o -o Main.x
[ese-wangy1@login03 fortran_demo1]$ ./Main.x
 19.4799995    15.7900000    19.2800007
 19.2800007    12.9200001    15.8599997
 15.8599997    11.2900000    14.0400000
 11.9300003    18.6000004    18.2299995
 19.2800007    12.9200001    15.8599997
 7.71999979    4.11000013    1.44000006    4.80000019    5.55000019
 5.55000019    4.80000019    4.03999996    0.589999974    8.57999992
 0.589999974    8.57999992    2.25999999    7.71999979    4.11000013

```

Output of MN.dat:

GNU nano 2.3.1					File: MN.dat
249.40	321.28	135.42	251.66	322.83	
229.90	277.34	115.80	222.61	283.04	
193.38	239.84	100.18	191.18	242.60	
206.09	294.73	133.52	208.97	300.72	
229.90	277.34	115.80	222.61	283.04	

PS6_2

2.1 Write a module Declination_angle:

```

module Declination_angle
implicit none

contains
  subroutine day_ang(day,ang)
    integer :: day
    real :: day2,ang
    day2=real(day)
    ang=-23.44*cos((day2+10)*360/365)
  end subroutine day_ang
end module Declination_angle

```

2.2 Write a module Solar_hour_angle:

```

module Solar_hour_angle
implicit none

contains
  subroutine cal_ang(lon,day,time,tz,ang)
    real :: lon,time,rad,eot,offset,lst
    integer :: day,tz
    real :: ang

    rad=2*3.14159/365*(day-1+(time-12)/24)
    eot=229.18*(0.000075+0.001868*cos(rad)-0.032077*sin(rad)-0.014615*cos(2*rad)-0.040849*sin(2*rad))
    offset=eot+4*(lon-15*tz)
    lst=time+offset/60
    ang=15*(lst-12)
  end subroutine cal_ang
end module Solar_hour_angle

```

2.3 Write a main program:

```

program Solar_elevation_angle
use Solar_hour_angle
use Declination_angle

implicit none
integer :: tz,day
real :: time,lon,ang

tz=8
day=365
lon=114.062996
time=10.53
call cal_ang(lon,day,time,tz,ang)
print*, ang

call day_ang(day,ang)
print*,ang
end program Solar_elevation_angle

```

2.4 Create a library:

```

[ese-wangy1@login03 fortran_demo1]$ gfortran -c Declination_angle.f90
[ese-wangy1@login03 fortran_demo1]$ gfortran -c Solar_hour_angle.f90

[ese-wangy1@login03 fortran_demo1]$ ar rcvf libsea.a Declination_angle.o Solar_hour_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o

```

Print the SEA for Shenzhen (The first line is the result of Solar_hour_angle, while the second line is the result of Declination_angle):

```

[ese-wangy1@login03 fortran_demo1]$ gfortran Solar_elevation_angle.f90 -o Solar_elevation_angle.x -L. -lsea
[ese-wangy1@login03 fortran_demo1]$ ./Solar_elevation_angle.x
-28.5934067
-15.5591497

```