学生：左小幸

批改：李娟

得分：37/40

# PS6

# 12132243 左小幸

## 1. Matrix multiplication

**1.1 [5 points]** Write a program `Main.f90` to read `fortran_demo1/M.dat` as the matrix `M`, and `fortran_demo1/N.dat`as the matrix `N`.

**1.2 [5 points]** Write a subroutine `Matrix_multip.f90` to do matrix multiplication.

**1.3 [5 points]** Call the subroutine `Matrix_multip()` from `Main.f90` to compute `M*N`; write the output to a new file `MN.dat`, values are in formats of `f9.2`.

```
program main

use Constants

implicit none

   real :: M(5,3),N(3,5),MN(5,5)

   ! opening file and reading
   open(1, file = 'M.dat')
   read(1,*), M
   close(1)

   open(2, file = 'N.dat')
   read(2,*), N
   close(2)

   call Matrix_multip(M,N,MN)

   open(unit=3, file='MN.dat')
   write(3,'(f9.2)'), MN
   close(3)

end program main

!---------------------------------------
```

This is the **main.f90** file

```fortran
module Constants
implicit none

 contains

 subroutine Matrix_multip(M,N,MN)
 implicit none

 real :: M(5,3),N(3,5),MN(5,5)
 MN = matmul(M,N)

 end subroutine Matrix_multip

end module Constants
~
```

This is the **Matrix_multip.f90** file contains the subroutine.

```
242.35
213.33
223.01
225.15
171.61
284.76
264.99
262.21
242.17
216.25
168.55
150.62
149.53
156.89
119.57
336.08
301.25
337.51
284.56
253.97
242.91
233.05
209.97
205.06
186.48
~
```

This is the result MN.dat

# 2. Calculate the Solar Elevation Angle

**2.1 [5 points]** Write a module `Declination_angle` that calculates the *declination angle* on a given date.

[**Hint:** using the “Better formula” from [Solar Declination Angle & How to Calculate it]]

```
module Declination_angle
implicit none

real, parameter :: pi = 3.1415926536

contains

 subroutine declination(date)

  character (len=8),intent(in) :: date
  integer :: year,month,day
  integer :: days
  real :: sda
  integer :: dayofmonth(12)=[31,28,31,30,31,30,31,31,30,31,30,31]

  common days,sda

 ! write(*,*)"Input the date(YYYYMMDD):"
 ! read(*,*)date

  read(date(1:4),*) year
  read(date(5:6),*) month
  read(date(7:8),*) day

  ! 判断是否闰年
  if(((MOD(year,4)==0).and.(MOD(year,100)/=0)).or.(mod(year,400)=
=0)) then
    dayofmonth(2)=29
  else
    dayofmonth(2)=28
  end if

  ! 计算天数
  days=sum(dayofmonth(1:month-1))+day

  ! 计算declination angle
  sda = asin(sin(-23.44*pi/180)*cos((360/365.24*(Days+10)+360/pi*
0.0167*sin((360/365.24*(Days-2))*pi/180))*pi/180))/pi*180

  ! write(*,*)'solar declination is:',sda

 end subroutine declination

!  real function


end module Declination_angle
```

**This is module `Declination_angle`**

**2.2 [10 points]** Write a module `Solar_hour_angle` that calculates the *solar hour angle* in a given location for a given date and time.

[**Hint:** using the formulas from Solar Hour Angle & How to Calculate it]

```
module Solar_hour_angle

implicit none

  real, parameter :: pi = 3.1415926536

contains

  subroutine Sha(lon,days,time)

    integer :: hour,minute
    integer,intent(in) :: days
    real,intent(in) :: lon
    real :: hours,gama,eot,offset,LST,dtz
    real :: h
    character(len=5),intent(in) :: time

    common h

    read(time(1:2),*) hour
    read(time(4:5),*) minute

    hours = hour+minute*1.0/60

    gama = 2*pi/365*(days-1+(hours-12)/24)
    eot = 229.18*(0.000075+0.001868*cos(gama)-0.032077*sin(gama)-
 0.014615*cos(2*gama)- 0.40849*sin(2*gama))
    dtz = ceiling((lon-7.5)/15)
    offset = eot+4*(lon-15*dtz)
    LST = LST+offset

    ! LST in hour
    h = 15*(LST-12)
    write(*,*) 'solar hour angle is',h

  end subroutine Sha

end module Solar_hour_angle
```

This is solar hour angle module

**2.3 [5 points]** Write a main program (`Solar_elevation_angle.f90`) that uses module `Declination_angle` and `Solar_hour_angle` to calculate and print the SEA in a given location for a given date and time.:

```
program solar_elevation_angle

use Declination_angle
use Solar_hour_angle

implicit none

real, parameter :: p = 3.1415926536
character(len=8) :: date
character(len=5) :: time
real :: lat,lon,h,sda
real :: sea
integer :: days

  write(*,*)"Input the date(YYYYMMDD):"
  read(*,*)date
  call declination(date)

  write(*,*)"Input the time(HH:MM):"
  read(*,*) time
  write(*,*)"Input the location:"
  write(*,*)"longitude:"
  read(*,*) lon
  write(*,*)"latitude:"
  read(*,*) lat
  call Sha(lon,days,time)

  sea = asin(sin(lat/180*p)*sin(sda/180*p)+cos(lat/180*p)*cos(sda
/180*p)*cos(h/180*p))/p*180
  write(*,*) 'solar elevation angle is:',sea


end program Solar_elevation_angle
~
```

This is the main programe.

```
[ese-zuoxx@login01 fortran_demo1]$ gfortran Solar_hour_angle.f90
Declination_angle.f90 Solar_elevation_angle.f90 -o Solar_elevatio
n_angle.x
```

Compile the programe file with two modules.

```
[ese-zuoxx@login01 fortran_demo1]$ ./Solar_elevation_angle.x
 Input the date(YYYYMMDD):
20211222
 Input the time(HH:MM):
17:00
 Input the location:
 longitude:
113.5
 latitude:
22.5
 solar hour angle is    68.4371262
 solar elevation angle is:    67.4999924
```

result

**2.4 [5 points]** Create a library (`libsea.a`) that contains `Declination_angle.o` and `Solar_hour_angle.o`. Compile `Solar_elevation_angle.f90` using `libsea.a`. Print the SEA for Shenzhen (`22.542883N, 114.062996E`) at `10:32`(Beijing time; UTC+8) on `2021-12-31`.

```
[ese-zuoxx@login01 fortran_demo1]$ gfortran -c Declination_angle.
f90
[ese-zuoxx@login01 fortran_demo1]$ gfortran -c Solar_hour_angle.f
90
```

```
[ese-zuoxx@login01 fortran_demo1]$ gfortran Solar_elevation_angle
.f90 Declination_angle.o Solar_hour_angle.o -o Solar_elevation_an
gle.x
[ese-zuoxx@login01 fortran_demo1]$ ar rcvf libsea.a Solar_hour_an
gle.o Declination_angle.o
a - Solar_hour_angle.o
a - Declination_angle.o
[ese-zuoxx@login01 fortran_demo1]$ ar tv libsea.a
rw-r--r-- 2224/1360   3784 Dec 22 18:00 2021 Solar_hour_angl
rw-r--r-- 2224/1360   3392 Dec 22 17:59 2021 Declination_ang
```

参考结果：

SHA=-28.43

SD=-23.13

SEA=36.61°，-3