CAI Kezhu (蔡可祝)'s TA report for assignment06
SID: 12132496
Github: https://github.com/Baicai0908/ESE5023_Assignments_12132496
Responsible TA: HUANG Hao
Grade: 39

1.1、

Good for 1 (15/15)

```
program Main

implicit none

integer :: i
real, dimension(5,3) :: M
real, dimension(3,5) :: N
open(unit = 1, file = "M.dat", status = 'old')

do i = 1,5
    read (1,*) M(i,1:3)
enddo

write(*,*) 'M:'
do i = 1,5
    write(*,"(*(f5.2,1X))") M(i,:)
end do

close(1)

open(unit = 2, file = "N.dat", status = 'old')

do i = 1,3
    read (2,*) N(i,1:5)
enddo

write(*,*) 'N:'

do i = 1,3
    write(*,"(*(f4.2,1X))") N(i,1:5)
enddo

close(2)
end program Main
```

```
[ese-caikzh@login01 fortran_demo1]$ nano Main.f90
[ese-caikzh@login01 fortran_demo1]$ gfortran Main.f90 -o Main.x
[ese-caikzh@login01 fortran_demo1]$ ./Main.x
 M:
19.48 15.79 19.28
19.28 12.92 15.86
15.86 11.29 14.04
11.93 18.60 18.23
19.28 12.92 15.86
 N:
7.72 4.11 1.44 4.80 5.55
5.55 4.80 4.04 0.59 8.58
0.59 8.58 2.26 7.72 4.11
```

1.2、

```
program Main

implicit none

integer :: i,j
real, dimension(5,3) :: M
real, dimension(3,5) :: N
real, dimension(5,5) :: R
open(unit = 1, file = "M.dat", status = 'old')

do i = 1,5
    read (1,*) M(i,1:3)
enddo

write(*,*) 'M:'
do i = 1,5
    write(*,"(*(f5.2,1X))") M(i,:)
end do

close(1)

open(unit = 2, file = "N.dat", status = 'old')

do i = 1,3
    read (2,*) N(i,1:5)
enddo

write(*,*) 'N:'

do i = 1,3
    write(*,"(*(f4.2,1X))") N(i,1:5)
enddo

close(2)

do i = 1,5
    do j = 1,5
        R(i,j) = 0
    enddo
enddo

call Matrix_multip(M,N,R)

write(*,*) 'R:'
do i = 1,5
    write(*,"(*(f6.2,1X))") R(i,:)
enddo
```

```
end program Main

subroutine Matrix_multip(M,N,R)

implicit none

integer :: i,j,k
real, dimension(5,3), intent(in) :: M
real, dimension(3,5), intent(in) :: N
real, dimension(5,5), intent(out) :: R

do i = 1,5
    do j = 1,5
        do k = 1,3
            R(i,j) = R(i,j)+M(i,k)*N(k,j)
        enddo
    enddo
enddo

end subroutine Matrix_multip
```

```
[ese-caikzh@login01 fortran_demo1]$ ./Main.x
 M:
19.48 15.79 19.28
19.28 12.92 15.86
15.86 11.29 14.04
11.93 18.60 18.23
19.28 12.92 15.86
 N:
7.72 4.11 1.44 4.80 5.55
5.55 4.80 4.04 0.59 8.58
0.59 8.58 2.26 7.72 4.11
 R:
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
```

1.3、

```
call Matrix_multip(M,N,R)

do i = 1,5
    open(unit = 3, file = 'MN.dat')
    write(3,'(*(f9.2))') R(i,:)
enddo

close(3)

end program Main
```

（其余代码如第二题所示）

```
[ese-caikzh@login03 fortran_demo1]$ nano 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
```

2.1、 I suggest you to use asind and sin, replacing asin(/pi*180) and sin(/180*pi).

```
module Declination_angle

implicit none

real, parameter :: pi = 3.1415926536
real :: da
integer, parameter :: d = 364

contains

subroutine dec_angle()

da = asin(sin(-23.44*pi/180)*cos(360/365.24*(d+10)*pi/180+2*0.0167*sin(360/365.24*(d-2))))

end subroutine dec_angle

end module Declination_angle
```

2.2、

```
module Solar_hour_angle

implicit none

real :: h, y, EoT, Offset, LST_new
real, parameter :: Long = 114.062996, LST = 10.53, pie = 3.1415926536
integer, parameter :: TZ = 8, d = 364

contains

subroutine sh_angle()
    y = 2*pie/365*(d-1+(LST-12)/24)
    EoT = 229.18*(0.000075+0.001868*cos(y)-0.032077*sin(y)-0.014615*cos(2*y)-0.040849*sin(2*y))
    Offset = EoT+4*(Long-15*TZ)
    LST_new = LST+Offset/60
    h = 15*(LST_new-12)
end subroutine sh_angle

end module Solar_hour_angle
```

2.3、

I did not receive your code through GitHub. I think you should call the two function before calculating sea.

And I think it indeed needs you to set parameters like year, date, hour, lat, lon, timezone, etc. as input of dec_angle(), and declination angle as output. So as the calculations of solar hour angle.

```
program Solar_elevation_angle

use Declination_angle

use Solar_hour_angle

implicit none

real, parameter :: fi = 22.542883
real :: sea

sea = asin(sin(fi*pi/180)*sin(da)+cos(fi*pi/180)*cos(da)*cos(h*pi/180))

call dec_angle()

call sh_angle()

write(*,*) 'Declination_angle:', da*180/pi
write(*,*) 'Solar_hour_angle:', h
write(*,*) 'Solar_elevation_angle:', sea*180/pi

end program Solar_elevation_angle
```

2.4、

```
[ese-caikzh@login03 fortran_demo1]$ gfortran -c Declination_angle.f90
[ese-caikzh@login03 fortran_demo1]$ gfortran -c Solar_hour_angle.f90
[ese-caikzh@login03 fortran_demo1]$ ar rcvf libsea.a Declination_angle.o Solar_hour_angle.o
r - Declination_angle.o
r - Solar_hour_angle.o
[ese-caikzh@login03 fortran_demo1]$ gfortran Solar_elevation_angle.f90 -o Solar_elevation_angle.x -L. -lsea
[ese-caikzh@login03 fortran_demo1]$ ./Solar_elevation_angle.x
 Declination_angle:  -23.2667198
 Solar_hour_angle:  -28.4798660
 Solar_elevation_angle:   67.4571152
```

网页上计算太阳高度角为 36.61°，我用计算器计算公式结果为 36.51°，相差不多，但是用 fortran 编译后不知道为啥变成了 67°多。

Only 1 point was deducted for the wrong answer.