Assignment 6

ESE 5023

张子严

12132873

量子科学与工程研究

2021年12月22日

1. Matrix Multiplication

The code is showed below.

```
[ese-zhangzy@login02 fortran_demo1]$ cat Main.f90
Program Main

implicit none

integer :: u1, u2, u3, i, j
real(4), dimension(:,:), allocatable :: a, b
real(4) :: c(5,5)

! read matrix from M.dat
u1 = 50
open(unit=u1, file='M.dat', status='old')
allocate(a(5,3))
do i=1, 5
    read(u1,*) (a(i,j), j=1,3)
enddo
print*,  "M:"
do i=1,5
    write(*,'(5f9.2)') (a(i,j), j=1,3)
enddo
close(u1)

! read matrix from N.dat
u2 = 51
open(unit=u2, file='N.dat', status='old')
allocate(b(3,5))
do i=1, 3
    read(u2, *) (b(i,j), j=1,5)
enddo
print*, "N:"
do i=1,3
    write(*, '(5f9.2)') (b(i,j), j=1,5)
enddo

close(u2)

! call the subroutine
call Matrix_multip(a,b,c)
```

```
! Print the return matrix
print*, "M*N="
write(*, '(5f9.2)') c

! write result to file MN.dat
u3 = 52
open(unit=u3, file='MN.dat', status='replace')
write(u3,'(5f9.2)') c
close(u3)

End Program Main
```

The running result is showed below:

```
[ese-zhangzy@login02 fortran_demo1]$ nano Main.f90
[ese-zhangzy@login02 fortran_demo1]$ gfortran Main.f90 Matrix_multip.f90 -o main.x
[ese-zhangzy@login02 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
 M*N=
   249.40   229.90   193.38   206.09   229.90
   321.28   277.34   239.84   294.73   277.34
   135.42   115.80   100.18   133.52   115.80
   251.66   222.61   191.18   208.97   222.61
   322.83   283.04   242.60   300.72   283.04
```

Here we check the requested files: Main.f90, Matrix_multip.f90 and MN.dat

```
[ese-zhangzy@login02 fortran_demo1]$ ls
a.out            ImplicitTypeTest.f90  MN.dat                 TestUndeclared.f90
DoLoopTest.f90   Implicit.x            N.dat                  test.x
DoWhileTest.f90  Main.f90              PrecisionTest.f90      VariableShowcase.f90
HelloWorld.f90   main.x                PrecisionTest.x        Variable.x
HelloWorld.x     Matrix_multip.f90     TestArray.f90
IfElseTest.f90   M.dat                 TestRelationalOps.f90
[ese-zhangzy@login02 fortran_demo1]$ cat MN.dat
   249.40   229.90   193.38   206.09   229.90
   321.28   277.34   239.84   294.73   277.34
   135.42   115.80   100.18   133.52   115.80
   251.66   222.61   191.18   208.97   222.61
   322.83   283.04   242.60   300.72   283.04
[ese-zhangzy@login02 fortran_demo1]$
```

2. Calculate the Solar Evaluation Angle

2.1 module Declination_angle

```
[ese-zhangzy@login02 fortran_demo1]$ cat Declination_angle.f90
module Declination_angle

implicit none

integer :: date
real(8) :: a,b,pi

contains
    subroutine get_angle()
        pi = 3.14159265
        print*, 'Input the number of days since January 1st'
        read(*,*) date

        a = COS(pi/180*(360/365.24)*(date+10)+(360/pi)*0.0167*SIN((pi/180*360/365.2
4)*(date-2)))
        b = (ASIN(SIN(-23.44*pi/180)*a))*180/pi

    end subroutine get_angle

end module Declination_angle
[ese-zhangzy@login02 fortran_demo1]$
```

2.2 module Solar_hour_angle

```
[ese-zhangzy@login02 fortran_demo1]$ cat Solar_hour_angle.f90
module Solar_hour_angle
    real(4) :: h,LST
    contains
    subroutine get_solar_angle()
        print*, 'Input the local solar time in the 24-hour format'
        read(*,*) LST

        h = 15 * ((LST / 60) -12)
    end subroutine get_solar_angle
end module Solar_hour_angle
[ese-zhangzy@login02 fortran_demo1]$
```

2.3 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.

```
[ese-zhangzy@login02 fortran_demo1]$ cat Solar_elevation_angle.f90
Program Solar_elevation_angle

use Declination_angle
use Solar_hour_angle

implicit none

real(4) :: sea, l

print*, 'Input latitude'
read(*,*) l

call get_angle()
call get_solar_angle()

sea = (ASIN(SIN(l*pi/180)*SIN(b*pi/180)+COS(l*pi/180)*COS(b*pi/180)*COS(h*pi/180)))
*180/pi

print*, 'Declination angle: ', b
print*, "Solar hour angle: ", h
print*, 'Solar elevation angle: ', sea

end program Solar_elevation_angle
```

2.4Create 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`.

Compile two modules:

```
[ese-zhangzy@login02 fortran_demo1]$ gfortran -c Solar_hour_angle.f90
[ese-zhangzy@login02 fortran_demo1]$ gfortran -c Solar_elevation_angle.f90
```

Place modules in the library libsea.a then compile,

```
[ese-zhangzy@login02 fortran_demo1]$ ar rcvf libsea.a Declination_angle.o Solar_hou
r_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o
[ese-zhangzy@login02 fortran_demo1]$ gfortran Solar_elevation_angle.f90 -o Solar_el
evation_angle_lib.x -L. -lsea
[ese-zhangzy@login02 fortran_demo1]$ ./Solar_elevation_angle_lib.x
```

Then run the program

```
[ese-zhangzy@login02 fortran_demo1]$ gfortran -c Solar_elevation_angle.f90
[ese-zhangzy@login02 fortran_demo1]$ ar rcvf libsea.a Declination_angle.o Solar_hou
r_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o
[ese-zhangzy@login02 fortran_demo1]$ gfortran Solar_elevation_angle.f90 -o Solar_el
evation_angle_lib.x -L. -lsea
[ese-zhangzy@login02 fortran_demo1]$ ./Solar_elevation_angle_lib.x
 Input latitude
34.5
 Input the number of days since January 1st
364
 Input the local solar time in the 24-hour format
600
 Declination angle:   -23.415861463273444
 Solar hour angle:   -30.0000000
 Solar elevation angle:    25.4577274
[ese-zhangzy@login02 fortran_demo1]$ s
```