GONG Guoqing (龚国庆)'s TA report for assignment06

SID: 12132198

Github: https://github.com/Gong001/ESE5023_Assignments_12132198

Responsible TA: HUANG Hao

Grade: 40

ESE5023 Assignment 6

12132198 Gong Guoqing

1. Matrix multiplication Good (15/15)

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.

Main.f90:

gong — ssh ese-gonggq@172.18.6.175 — 152×40

```fortran
program Main

implicit none

integer                               :: a, b, mcol, mrol, ncol, nrol, i, j
real(8),dimension(:,:),allocatable    :: M, N

a=50
b=51
mcol=3
mrol=4
ncol=4
nrol=3

open(unit=a,file='M.dat',status='old')
open(unit=b,file='N.dat',status='old')
allocate(M(mrol,mcol))
allocate(N(nrol,ncol))
do i=1,mrol
  read(a,*) M(i,:)
enddo

do i=1,nrol
  read(b,*) N(i,:)
enddo

do i=1,mrol
  write(*,*) "Line",i,":",M(i,:)
enddo
do i=1,nrol
  write(*,*) "Line ",i,":",N(i,:)
enddo

deallocate(M)
deallocate(N)
End Program Main
```

1,0-1 顶端

Output:

```
[ese-gonggq@login02 fortran_demo1]$ ./Main.x
 Line           1 :   19.480000000000000        15.789999999999999        19.280000000000001
 Line           2 :   19.280000000000001        12.920000000000000        15.859999999999999
 Line           3 :   15.859999999999999        11.289999999999999        14.039999999999999
 Line           4 :   11.930000000000000        18.600000000000001        18.230000000000000
 Line            1 :   7.7199999999999998        4.1100000000000003        1.4399999999999999        4.7999999999999998
 Line            2 :   5.5499999999999998        4.7999999999999998        4.0400000000000000       0.58999999999999997
 Line            3 :  0.58999999999999997        8.5800000000000001        2.2599999999999998        7.7199999999999998
[ese-gonggq@login02 fortran_demo1]$
```

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

Matrix_multip.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×29
subroutine Matrix_multip(M,N,MN)


implicit none

real(8),dimension(4,3),intent(in)      :: M
real(8),dimension(3,4),intent(in)      :: N
real(8),dimension(4,4),intent(out)     :: MN
integer                                :: i,j,k
real(8)                                :: t



do i=1,4
  do j=1,4
    t=0
    do k=1,3
      t=t+M(i,k)*N(k,j)
    enddo
    MN(i,j)=t
  enddo
enddo

end subroutine Matrix_multip
~
~
~
~
"Matrix_multip.f90" 24L, 420C                                    1,1           全部
```

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.

Main_read.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×51
Program Main_read
implicit none
integer :: a, b, mcol, mrol, ncol, nrol, i, j
real(8), dimension(:,:),allocatable :: M, N
real(8), dimension(4,4) :: MN
a=50
b=51
mcol=3
mrol=4
ncol=4
nrol=3
open(unit=a,file='M.dat',status='old')
open(unit=b,file='N.dat',status='old')

allocate(M(mrol,mcol))
allocate(N(nrol,ncol))

do i=1,mrol
read(a,*) M(i,:)
enddo

do i=1,nrol
read(b,*) N(i,:)
enddo

close(a)
close(b)

do i=1,mrol
write(*,*) "Line ",i,":",M(i,:)
enddo

do i=1,nrol
write(*,*) "Line ",i,":",N(i,:)
enddo

call Matrix_multip(M,N,MN)
do i=1,4
write(*,*) "Line ",i,":",MN(i,:)
enddo
open(unit=a,file='new1.dat',status='replace')
do i=1,4
write(a,'(f9.2)') MN(i,:)
enddo

close(a)
deallocate(M)
deallocate(N)
End Program Main_read
~
"Main_read.f90" 49L, 765C                                        41,1          全部
```

Output:

```
[ese-gonggq@login02 fortran_demo1]$ vi Main_read.f90
[ese-gonggq@login02 fortran_demo1]$ gfortran Main_read.f90 Matrix_multip.f90 -o Main_read.x
[ese-gonggq@login02 fortran_demo1]$ ./Main_read.x
 Line           1 :   19.480000000000000        15.789999999999999        19.280000000000001
 Line           2 :   19.280000000000001        12.920000000000000        15.859999999999999
 Line           3 :   15.859999999999999        11.289999999999999        14.039999999999999
 Line           4 :   11.930000000000000        18.600000000000001        18.230000000000000
 Line           1 :   7.7199999999999998        4.1100000000000003        1.4399999999999999        4.7999999999999998
 Line           2 :   5.5499999999999998        4.7999999999999998        4.0400000000000000       0.58999999999999997
 Line           3 :  0.58999999999999997       8.5800000000000001        2.2599999999999998        7.7199999999999998
 Line           1 :   249.39530000000002        321.27719999999999        135.41559999999998        251.66170000000000
 Line           2 :   229.90499999999997        277.33560000000000        115.80360000000000        222.60599999999999
 Line           3 :   193.38229999999999        239.83980000000000        100.18039999999999        191.17789999999999
 Line           4 :   206.08529999999999        294.72569999999996        133.52300000000000        208.97360000000000
```

vi new1.dat: output

```
gong — ssh ese-gonggq@172.18.6.175 — 152×51
249.40
321.28
135.42
251.66
229.90
277.34
115.80
222.61
193.38
239.84
100.18
191.18
206.09
294.73
133.52
208.97
~
~
~
```

2. Calculate the Solar Elevation Angle (25/25)

2.1 [5 points] Write a module Declination_angle that calculates the *declination angle* on a given date. As I wrote for Penghan, I suggest you to use asind and sin, replacing asin(/pi*180) and sin(/180*pi).

Declination_angle.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 129×24
module Declination_angle

implicit none

real, parameter         :: pi=3.1415926536

contains

  subroutine cal_angle(m,d,da)

  implicit none

  integer,intent(in)            :: m, d
  real(8),intent(out)           :: da
  integer                       ::doy

  doy=(m-1)*30+d
  da=asin(sin(-23.44/180*pi)*cos(((360/365.24)*(doy+10)+360/pi*0.0167*sin(360/365.24*(doy-2)))/180*pi))
  da=da/pi*180

  end subroutine cal_angle
end module Declination_angle

                                                                                                            1,1            全部
```

Date.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 129×24
Program Date

use Declination_angle

implicit none

real(8)                 ::angle
integer                 ::m, d

m=12
d=18

call cal_angle(m,d,angle)

write(*,*) angle

end program Date
~
~
~
~
~
~
"Date.f90" 17L, 189C                                                                                        17,1           全部
```

Output:

```
[ese-gonggq@login02 newdir]$ vi Declination_angle.f90
[ese-gonggq@login02 newdir]$ vi Date.f90
[ese-gonggq@login02 newdir]$ gfortran Date.f90 Declination_angle.f90 -o Date.x
[ese-gonggq@login02 newdir]$ ./Date.x
  -23.335786319814492
```

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.

Solar_hour_angle.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×34
module Solarhourangle

implicit none

real, parameter          :: pi=3.1415926536

contains

  subroutine cal_sha(lon,m,d,t,sha)

  implicit none


  integer,intent(in)              :: m, d
  real(8),intent(in)              :: lon, t
  real(8),intent(out)             :: sha
  integer                         :: doy
  real(8)                         :: offset, eot, gam
  doy=(m-1)*30+d
  gam=2*pi/365*(doy-1+(t-12)/24)
  eot=229.18*(0.000075+0.001868*cos(gam)-0.032077*sin(gam)-0.014615*cos(2*gam)-0.040849*sin(2*gam))
  offset=eot+MOD(lon,15.0)
  sha=15*(t-12)+offset/60

  end subroutine cal_sha

end module Solarhourangle

~
~
~
~
~
"Solar_hour_angle.f90" 28L, 621C                                                    18,3          全部
```

Location.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×35
program location

use Solarhourangle

implicit none

real(8)          :: t,lon,h
integer          :: m,d

t=18
lon=118.24
m=12
d=18


call cal_sha(lon,m,d,t,h)

write(*,*) h

end program location
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"location.f90" 20L, 194C                                                            13,1          全部
```

Output:

```
[ese-gonggq@login02 newdir]$ gfortran location.f90 Solar_hour_angle.f90 -o location.x
[ese-gonggq@login02 newdir]$ ./location.x
   90.308419527005640
```

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.

Solar_elevation_angle.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×35
program Solar_elevation_angle

use Declination_angle
use Solarhourangle

implicit none

real, parameter         :: pai=3.1415926536
real(8)                 :: lat,lon,t,sha,da
integer                 :: m,d
real(8)                 :: sea

lat=32.22
lon=1.0
t=10.0
m=3
d=3

call cal_angle(m,d,da)
call cal_sha(lon,m,d,t,sha)

sea=asin(sin(lat/180*pai)*sin(da/180*pai)+cos(lat/180*pai)*cos(da/180*pai)*cos(sha/180*pai))
sea=sea/pai*180.0


write(*,*) sea

end program Solar_elevation_angle
~
~
~
~
~
~
"Solar_elevation_angle.f90" 28L, 488C                                                    28,1          全部
```

Output:

```
[ese-gonggq@login02 newdir]$ gfortran Solar_elevation_angle.f90 Declination_angle.f90 Solar_hour_angle.f90 -o Solar_elevation_angle.x
[ese-gonggq@login02 newdir]$ ./Solar_elevation_angle.x
   41.045703954998608
```

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.

shenzhen.f90:

```
gong — ssh ese-gonggq@172.18.6.175 — 142×35

program shenzhen

use Declination_angle
use Solarhourangle

implicit none

real, parameter         :: pai=3.1415926536
real(8)                 :: lat,lon,t,sha,da
integer                 :: m,d
real(8)                 :: SEA

lat=22.542883
lon=114.062996
t=10.0+32/60
m=12
d=31

call cal_angle(m,d,da)
call cal_sha(lon,m,d,t,sha)

SEA=asin(sin(lat/180*pai)*sin(da/180*pai)+cos(lat/180*pai)*cos(da/180*pai)*cos(sha/180*pai))
SEA=SEA/pai*180.0

write(*,*) SEA

end program shenzhen
~
~
~
~
~
~
"shenzhen.f90" 28L, 481C                                                                 10,1          全部
```

Output:

```
[ese-gonggq@login02 newdir]$ vi shenzhen.f90
[ese-gonggq@login02 newdir]$ gfortran -c Declination_angle.f90
[ese-gonggq@login02 newdir]$ gfortran -c Solar_hour_angle.f90
[ese-gonggq@login02 newdir]$ ar rcvf libsea.a Declination_angle.o Solar_hour_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o
[ese-gonggq@login02 newdir]$ gfortran shenzhen.f90 -o shenzhen.x -L. -lsea
[ese-gonggq@login02 newdir]$ ./shenzhen.x
   35.790305803209272
[ese-gonggq@login02 newdir]$
```

##

Thank for a clear citation.

In 1.1, Peng Han explained to me the use of "allocate".
In 1.2, Peng Han explained to me how to create a "subroutine".
In 2.1, Peng Han explained to me the calculation of "declination angle" and the call of module.
In 2.4, Peng Han explained libsea.a to me.
Thanks a lot!