CAO Zhe (曹喆)'s TA report for assignment06
SID: 12132834
Github: https://github.com/kedaobilianxu/ESE5023_Assignments_12132834
Responsible TA: HUANG Hao
Grade: 40+1=41

Assignment 6

The code and report are relatively clear, and you validate your data by comparing them with websites's calculation results. Add 1 point for you.

12132834 曹喆

1. Matrix multiplication

Good (15/15)

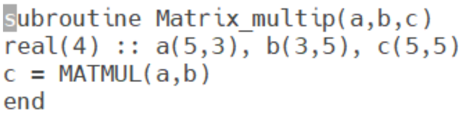
1.1 [5 points] Write a program `Main.f90` to read `fortran_demo1/M.dat` as the matrix `M`, and `fortran_demo1/N.dat` as the matrix `N`.

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`.

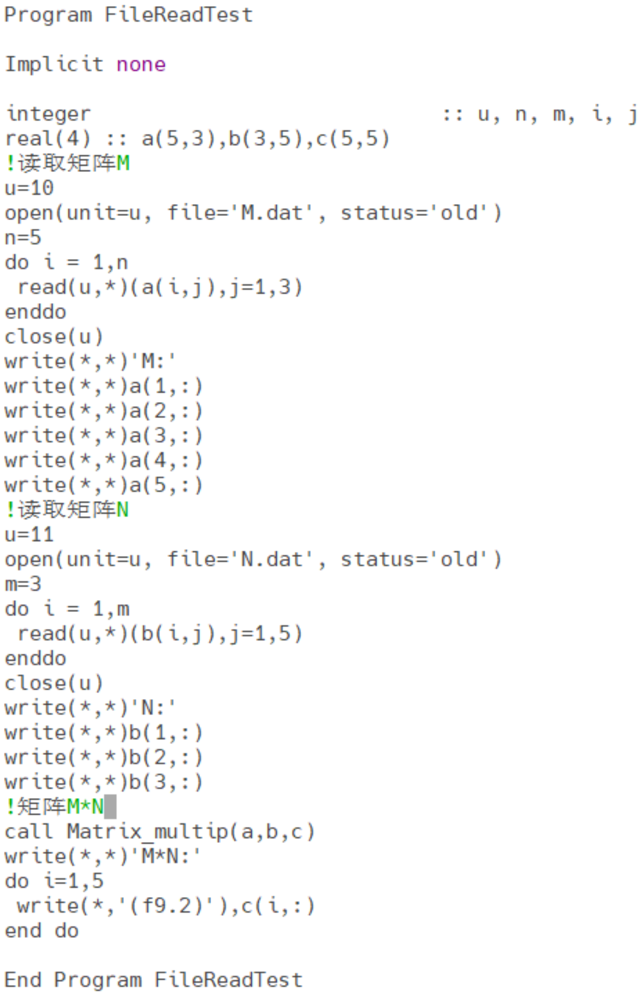
You did not write MN in a new file MN.dat. Instead, you print it by write(*,*). However, no point was deducted.

Matrix_multip.f90 代码：



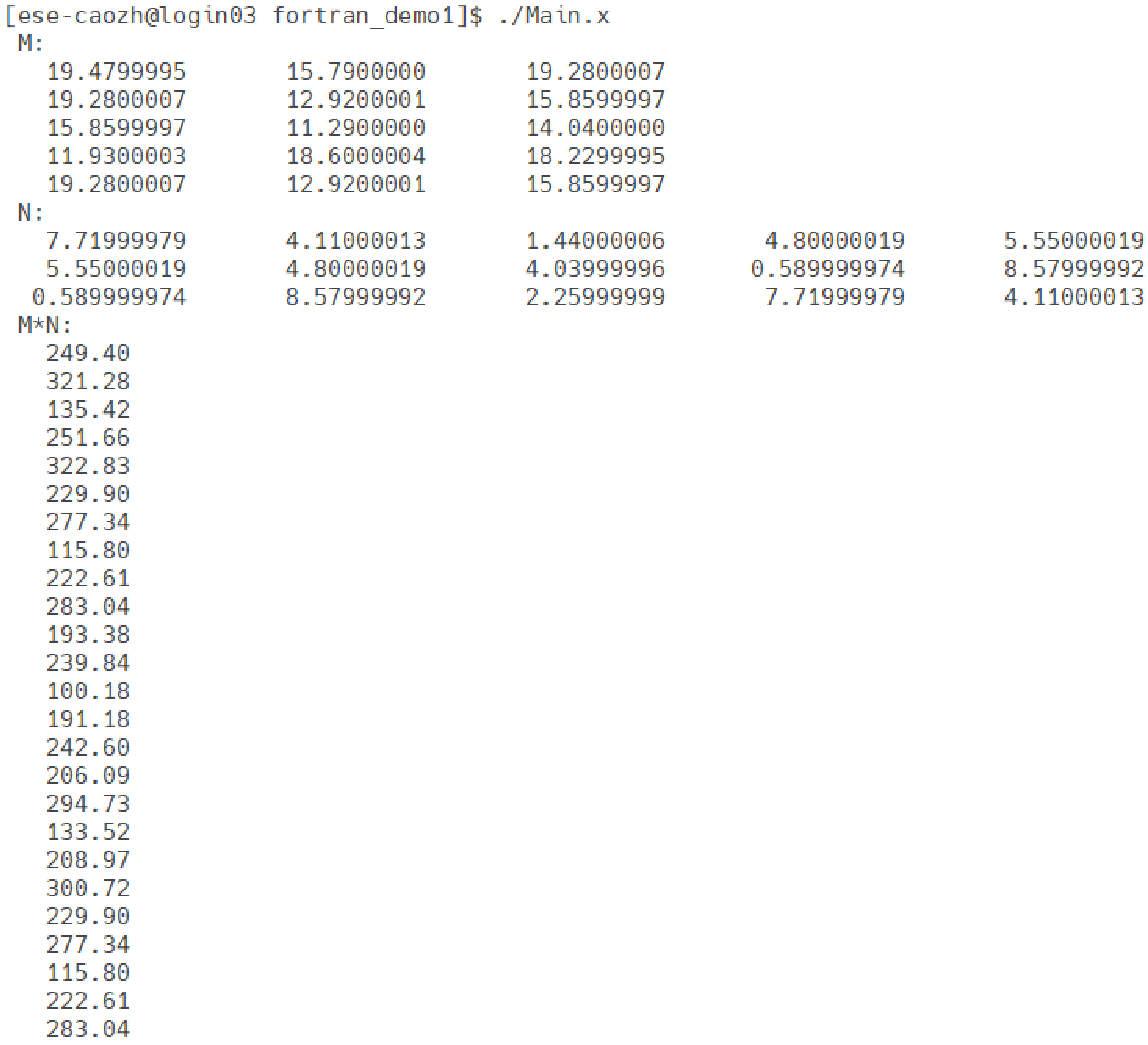
```
subroutine Matrix_multip(a,b,c)
real(4) :: a(5,3), b(3,5), c(5,5)
c = MATMUL(a,b)
end
```

Main.f90 代码：



```
Program FileReadTest

Implicit none

integer                         :: u, n, m, i, j
real(4) :: a(5,3),b(3,5),c(5,5)
!读取矩阵M
u=10
open(unit=u, file='M.dat', status='old')
n=5
do i = 1,n
 read(u,*)(a(i,j),j=1,3)
enddo
close(u)
write(*,*)'M:'
write(*,*)a(1,:)
write(*,*)a(2,:)
write(*,*)a(3,:)
write(*,*)a(4,:)
write(*,*)a(5,:)
!读取矩阵N
u=11
open(unit=u, file='N.dat', status='old')
m=3
do i = 1,m
 read(u,*)(b(i,j),j=1,5)
enddo
close(u)
write(*,*)'N:'
write(*,*)b(1,:)
write(*,*)b(2,:)
write(*,*)b(3,:)
!矩阵M*N
call Matrix_multip(a,b,c)
write(*,*)'M*N:'
do i=1,5
 write(*,'(f9.2)'),c(i,:)
end do

End Program FileReadTest
```

运行结果：



```
[ese-caozh@login03 fortran_demo1]$ ./Main.x
 M:
   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
 N:
   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
 M*N:
   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
```

说明：

因为 1.3 要求输出要保留两位小数，所以用 write(*,'(f9.2)')，但是这样输出不知道为何矩阵的每个值占输出的一列。

For this, you can use write(c, '(5f9.2)') MN(i, :), replacing write(c, '(f9.2)') MN(i, :), when you write MN.dat. Then, you can get 5×5 matrix

2. Calculate the Solar Elevation Angle

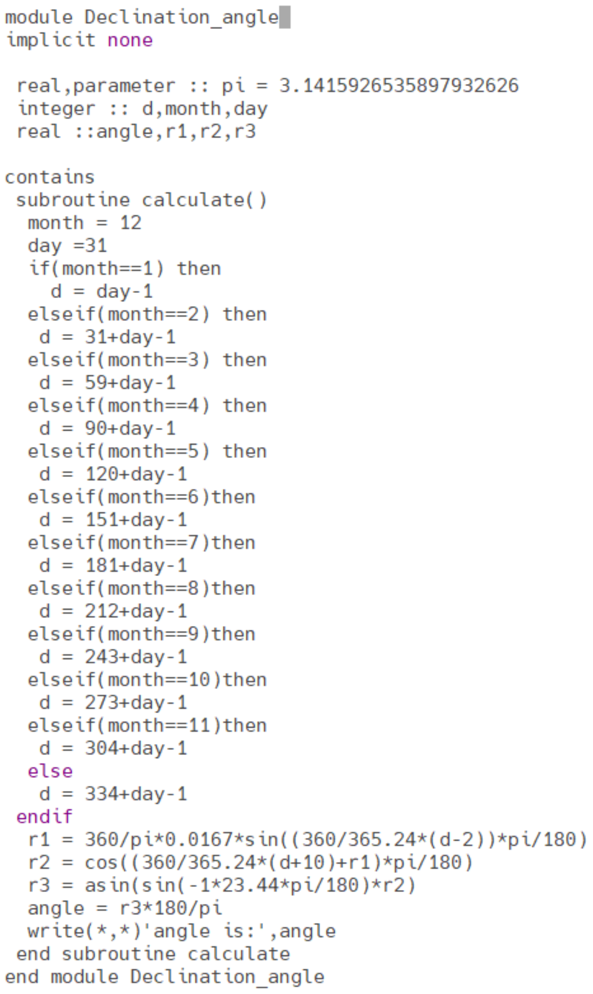
The solar elevation angle (SEA) is the angle between the imaginary horizontal plane on which you are standing and the sun in the sky. SEA is very important in deciding the inclination of solar panels, in both photovoltaics (PV) and thermal. The value of the SEA depends on the location on the Earth and the local date and time.

Please read this Solar Elevation Angle – Calculating Altitude of Sun and links therein for how to calculate SEA.

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]

代码：

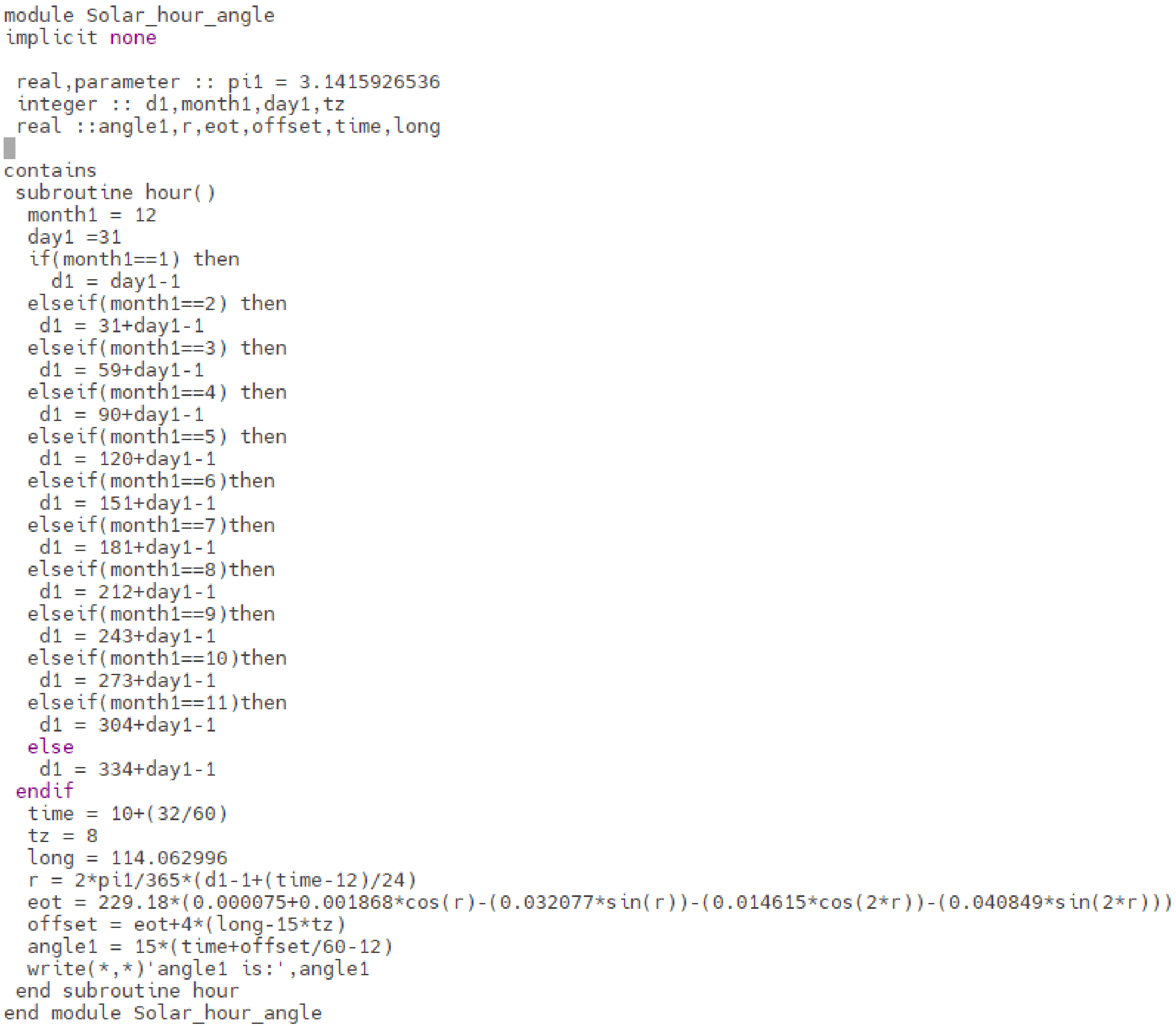


```
module Declination_angle
implicit none

 real,parameter :: pi = 3.1415926535897932626
 integer :: d,month,day
 real ::angle,r1,r2,r3

contains
 subroutine calculate()
  month = 12
  day =31
  if(month==1) then
    d = day-1
  elseif(month==2) then
   d = 31+day-1
  elseif(month==3) then
   d = 59+day-1
  elseif(month==4) then
   d = 90+day-1
  elseif(month==5) then
   d = 120+day-1
  elseif(month==6)then
   d = 151+day-1
  elseif(month==7)then
   d = 181+day-1
  elseif(month==8)then
   d = 212+day-1
  elseif(month==9)then
   d = 243+day-1
  elseif(month==10)then
   d = 273+day-1
  elseif(month==11)then
   d = 304+day-1
  else
   d = 334+day-1
 endif
  r1 = 360/pi*0.0167*sin((360/365.24*(d-2))*pi/180)
  r2 = cos((360/365.24*(d+10)+r1)*pi/180)
  r3 = asin(sin(-1*23.44*pi/180)*r2)
  angle = r3*180/pi
  write(*,*)'angle is:',angle
 end subroutine calculate
end 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]

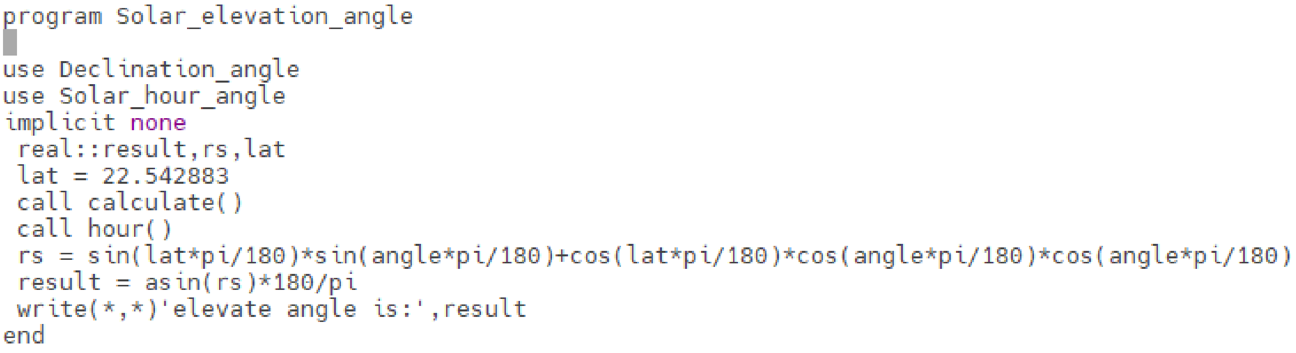
代码：



```
module Solar_hour_angle
implicit none

 real,parameter :: pi1 = 3.1415926536
 integer :: d1,month1,day1,tz
 real ::angle1,r,eot,offset,time,long

contains
 subroutine hour()
  month1 = 12
  day1 =31
  if(month1==1) then
    d1 = day1-1
  elseif(month1==2) then
   d1 = 31+day1-1
  elseif(month1==3) then
   d1 = 59+day1-1
  elseif(month1==4) then
   d1 = 90+day1-1
  elseif(month1==5) then
   d1 = 120+day1-1
  elseif(month1==6)then
   d1 = 151+day1-1
  elseif(month1==7)then
   d1 = 181+day1-1
  elseif(month1==8)then
   d1 = 212+day1-1
  elseif(month1==9)then
   d1 = 243+day1-1
  elseif(month1==10)then
   d1 = 273+day1-1
  elseif(month1==11)then
   d1 = 304+day1-1
  else
   d1 = 334+day1-1
 endif
  time = 10+(32/60)
  tz = 8
  long = 114.062996
  r = 2*pi1/365*(d1-1+(time-12)/24)
  eot = 229.18*(0.000075+0.001868*cos(r)-(0.032077*sin(r))-(0.014615*cos(2*r))-(0.040849*sin(2*r)))
  offset = eot+4*(long-15*tz)
  angle1 = 15*(time+offset/60-12)
  write(*,*)'angle1 is:',angle1
 end subroutine hour
end module Solar_hour_angle
```

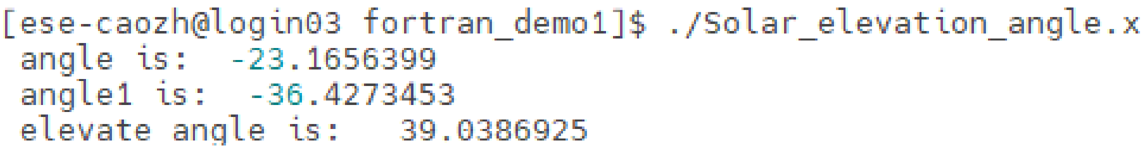
2.3 [5 points] Write a main program (`Solar_elevation_angle.f90`) that uses module `Declination_angle` and `Solar_hour_angle` to calculate and print the SEA in a given location for a given date and time.

代码：



```
program Solar_elevation_angle

use Declination_angle
use Solar_hour_angle
implicit none
 real::result,rs,lat
 lat = 22.542883
 call calculate()
 call hour()
 rs = sin(lat*pi/180)*sin(angle*pi/180)+cos(lat*pi/180)*cos(angle*pi/180)*cos(angle*pi/180)
 result = asin(rs)*180/pi
 write(*,*)'elevate angle is:',result
end
```

运行结果：



```
[ese-caozh@login03 fortran_demo1]$ ./Solar_elevation_angle.x
 angle is:  -23.1656399
 angle1 is:  -36.4273453
 elevate angle is:   39.0386925
```

说明：

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

2.1 说明：

Fortran 中的 sin(x)函数，x 是弧度，但是算出的结果是角度，所以需要角度转弧度。

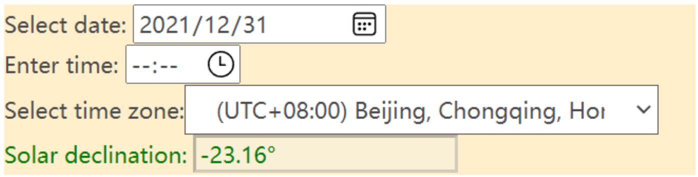
2.2 说明：

这里的带入三角函数的值都是弧度，所以不需要转化。

2.3 说明：

所计算的数据是深圳(22.542883N,114.062996E) 在 2021-12-31 10:32(UTC+8) 的 `Declination angle`、`solar hour angle` 和 `elevation angle`

题目所给网站计算出的 `Declination angle` 如下：



Select date: 2021/12/31

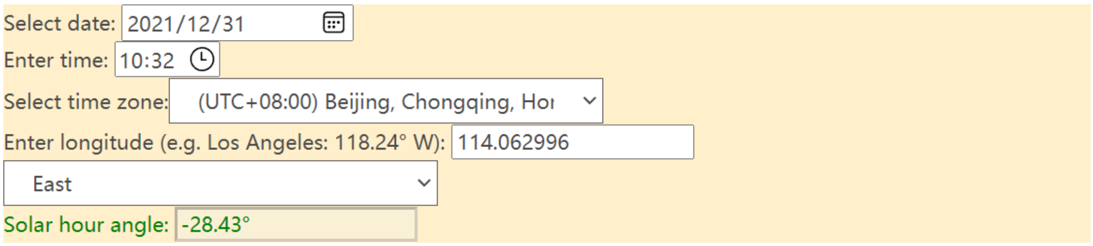
Enter time: --:--

Select time zone: (UTC+08:00) Beijing, Chongqing, Hor

Solar declination: -23.16°

可以看出：程序计算的值为-23.1656399,与计算器计算的接近。

题目所给网站计算出的 `solar hour angle` 如下：



Select date: 2021/12/31

Enter time: 10:32

Select time zone: (UTC+08:00) Beijing, Chongqing, Hor

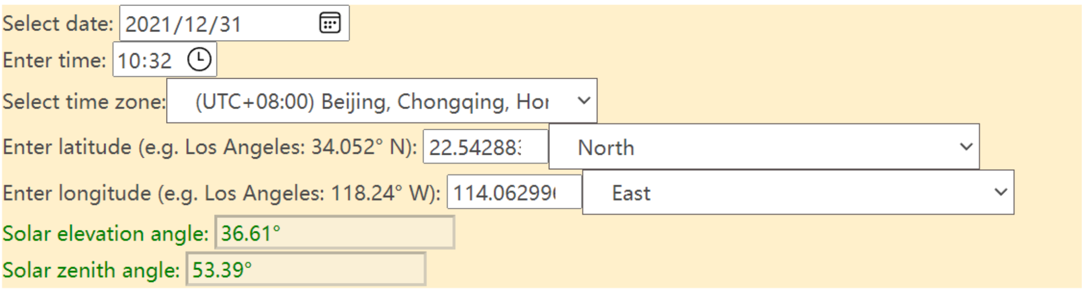
Enter longitude (e.g. Los Angeles: 118.24° W): 114.062996

East

Solar hour angle: -28.43°

可以看出：程序计算的值为-36.4273453,与计算器计算的差距较大，经过调查发现，计算器在计算角度的每个步骤都只保留小数点后两位，在程序中每步都是单精度浮点数，所以每一步都造成差距，但是我不太清楚如何在计算的过程中保留两位小数，也没有查到。

题目所给网站计算出的 solar elevation angle 如下：



Select date: 2021/12/31

Enter time: 10:32

Select time zone: (UTC+08:00) Beijing, Chongqing, Hoı

Enter latitude (e.g. Los Angeles: 34.052° N): 22.54288 North

Enter longitude (e.g. Los Angeles: 118.24° W): 114.06299 East

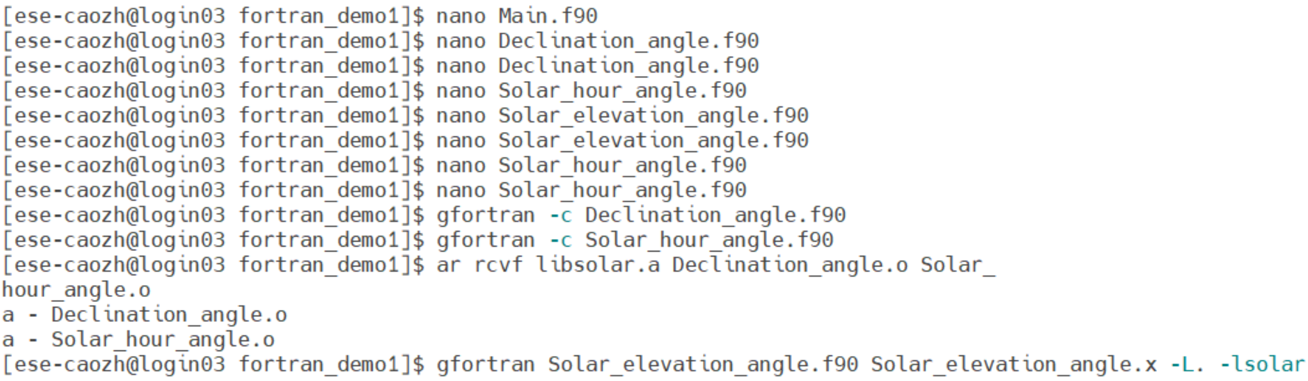
Solar elevation angle: 36.61°

Solar zenith angle: 53.39°

可以看出：程序计算的值为 39.0386925，与计算器计算出的差距不大，误差还是在计算 solar hour angle 上。

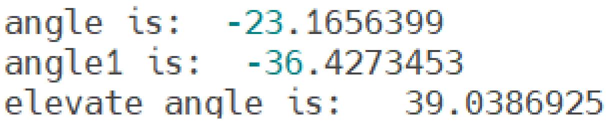
2.4[5points] Createalibrary(`libsea.a`)thatcontains `Declination_angle.o` and `Solar_hour_angle.o`.Compile `Solar_elevation_angle.f90` using `libsolar.a`. Print the SEA for Shenzhen (`22.542883N, 114.062996E`) at `10:32` (Beijing time; UTC+8) on `2021-12-31`.

过程：



```
[ese-caozh@login03 fortran_demo1]$ nano Main.f90
[ese-caozh@login03 fortran_demo1]$ nano Declination_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Declination_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Solar_hour_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Solar_elevation_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Solar_elevation_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Solar_hour_angle.f90
[ese-caozh@login03 fortran_demo1]$ nano Solar_hour_angle.f90
[ese-caozh@login03 fortran_demo1]$ gfortran -c Declination_angle.f90
[ese-caozh@login03 fortran_demo1]$ gfortran -c Solar_hour_angle.f90
[ese-caozh@login03 fortran_demo1]$ ar rcvf libsolar.a Declination_angle.o Solar_
hour_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o
[ese-caozh@login03 fortran_demo1]$ gfortran Solar_elevation_angle.f90 Solar_elevation_angle.x -L. -lsolar
```

运行结果：



```
 angle is:   -23.1656399
 angle1 is:   -36.4273453
 elevate angle is:    39.0386925
```