```
! Name:           Jared Doster and Chiel Donkers
! Course:       PHY 832
! Project:     Monte Carlo Ising Model: Wolf Algorithm and Metropolis Test
!
!Program Summary (Ising)
!
!    Input:       Hardcode three parameters to the subroutine (temperature of electrons
!                   and the length and width of the eletron array).
!
!    Process:   Creates an array representing the arry of electrons.
!                   Calls the subroutine in a nested loop from temp=0 to tempfinal=arbitrary.
!                   Each iteration of the outer "temperature" loop returns the magnetizaton for a
!                   particular temperature.
!                   The iterations of the inner loop will continue until the magnetization returned
!                   from the subroutine has converged to within a tolerance range
!
!    Output:      Stores the converged magnetization as a function of temperature for each iteration
!                   of the outer "temperature" loop
!
!
!Subroutine Summary (Mainloop)
!
! Input:         Three parameters from the program
!
! Process:     For a particular temperature, runs a loop involving random selection of electrions
!                   and performing a Metropolis Test. Each time that the mainloop is called, the spins
!                   of the array are randomly chosen and flipped.
!                   Many iterations of this loop cause the calculated magnetization to converge to a range
!                   of values.
!
! Output:       Magnetization for a particular temperature for a particular iteration.
!                   Output will vary each time that the subroutine is called



program ising

   implicit none



!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Declarations !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!! INPUT: Final temperature (Kelvin), final time, row and column size, step of temperature loop
   integer,parameter :: tempfinal = 40         ! Final temperature at end of temperature loop
   integer,parameter :: timefinal = 50000        ! Final time at end of time loop
   integer,parameter :: step = 1
   integer,parameter :: rowsize = 20
   integer,parameter :: colsize = 20
```

```
!! fortran begins indexing from 1. Start it from 0 because the rand() starts from 0
   real, dimension(0:rowsize-1, 0:colsize-1) :: spin

   real :: mag
   integer :: cputime, tempcount, timecount



!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Main Body !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


!!! Open files for writing data !!!
   call opentextfiles


!! Initialize Configuration (spin up = 1, spin down = -1)

!!    spin = 1    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! REMOVE


!! Seed random number generator

   call system_clock(cputime)
   call srand(cputime)


!! Run main loop subroutine
!! In the following nested loop, the outer loop is the temperature loop (for plot of magnetization vs.
temperature)
!! and the inner loop is the converging time loop (for plot of magnetization vs. time)


   tempcount=0
   do while (tempcount .LE. tempfinal)

! Re-initialize the spin lattice for every temperature
      spin=1

      timecount=0
      do while (timecount .LE. timefinal)
            call mainloop(spin, rowsize, colsize, tempcount, mag, timecount)
```

```
! We want time to print only once (choose an arbitrary temperature)
            if (tempcount == 25) then
                WRITE(16,*) mag
            endif


            timecount = timecount + step
        enddo


        WRITE(15,*) tempcount, abs(mag)


        tempcount = tempcount+1
   enddo





!!! Close text files !!!
   call closetextfiles



end program ising



!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Subroutines !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!-------------------------------------------------------------------------------------------------
!! Mainloop

subroutine mainloop(spin, rowsize, colsize, tempcount, mag, timecount)

   implicit none

!! Passed parameters, intent(in) parameters cannot be altered

   integer,intent(in) :: rowsize
   integer,intent(in) :: colsize
   integer,intent(in) :: tempcount
   integer,intent(in) :: timecount !! can remove later!!!!!!!!!!!
   real,dimension(0:rowsize-1, 0:colsize-1) :: spin
   real :: mag



!! Subroutine declerations
   integer :: i,j,ix,iy
```

```
   real :: temp
   real :: totup=0
   real :: totdown=0
   real :: r1,r2,r3
   real :: expo, ediff, enew=0, eold=0
   real :: oldsp, newsp, st, sb, sl, sr, neighbors


!! Tempcount is an integer, we want to rescale it to a real

   temp = tempcount/10.


!! Randomly choose a location in the array (the old spin)
!! 在二维结构中任取一个点，它的坐标为(ix, iy)

   r1 = rand()
   r2 = rand()

   ix = (r1*rowsize)-1
   iy = (r2*colsize)-1

!! 在这个的自旋为 spin(ix, iy) = 1 ，  -1

   oldsp = spin(ix,iy)

!! Calculate energy due to the neighbors (the if-statements takes into account the free boundaries)
!! 确定这个点周围四个近邻点的值。

   sl = spin(ix-1, iy)
   sr = spin(ix+1, iy)
   st = spin(ix, iy-1)
   sb = spin(ix, iy+1)

   if (ix .EQ. 0) sl = 0
   if (ix .EQ. rowsize) sr = 0
   if (iy .EQ. 0) st = 0
   if (iy .EQ. colsize) sb = 0

!! 计算能量，因为是 ising 模型，所以能量为  sigma_i sigma_j，即它们的乘机。
   neighbors = -sl - sr - st - sb

!! Calculate energy of old spin
```

```
   eold = oldsp*neighbors

!    if (oldsp .EQ. 1) then
!         eold = 1
!    elseif (oldsp .EQ. -1) then
!         eold = -1
!    else
!         print*, 'At least one oldsp is neither up nor down.     ', oldsp, temp, timecount
!    endif


!! 翻转自旋，能量也变号。
!! Choose new spin & calculate energy of new spin

   enew = -eold
   newsp = -oldsp



!! Calculate energy difference between old and new spins
!! 计算能量差
   ediff = (enew - eold)




!! Calculate energy and exponential
!! 抽样。

   expo = exp(-ediff/temp)



!! Metropolis test

   r3 = rand()

   if (expo > r3) then
      spin(ix,iy) = newsp
   endif



!! Calculate magnetization    (quantifies how magnetic the material is)
!! (first initialize counters)

   totup = 0
   totdown = 0

   do i=0,rowsize-1
```

```
      do j=0,colsize-1

         if (spin(i,j) == 1) then
            totup = totup + 1
         elseif (spin(i,j) == -1) then
              totdown = totdown + 1
         else
              print*, 'At least one spin is neither up nor down'
         endif

       end do
   end do

   mag = (totup - totdown)/(rowsize*colsize)

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!! Make average of last 20 magnetizations
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!



end subroutine mainloop



!-------------------------------------------------------------------------------
!! Open data files

subroutine opentextfiles
      integer :: OPEN_STATUS
      OPEN(UNIT=15,FILE="mag_temp.txt",STATUS="REPLACE",IOSTAT=OPEN_STATUS)
      if (OPEN_STATUS /= 0) then
           STOP "------------Error, mag_temp file not opened properly------------"
      endif
      OPEN(UNIT=16,FILE="mag_time.txt",STATUS="REPLACE",IOSTAT=OPEN_STATUS)
      if (OPEN_STATUS /= 0) then
           STOP "------------Error, mag_time file not opened properly------------"
      endif



end subroutine



!-------------------------------------------------------------------------------
!! Close data files

subroutine closetextfiles
      CLOSE(UNIT=15)
      CLOSE(UNIT=16)
endsubroutine
```