```
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! This code gives us the diagonal elements of an Ising model for a
! spin-1/2 chain of size L and a certain fixed number of up-spins
!
! In the code: L is called "chain", the dimension is "dimTotal", and the
! site-basis vectors are "basis".
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!             VARIABLES to be used in the whole code
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

       module variables

       implicit none

       integer (kind=4) :: i,j,ii
       integer (kind=4) :: chain,upspins,dimTotal
       integer (kind=4) :: facL,facU,facLU

! PARAMETERS of the ISING Hamiltonian
       real (kind=8) :: JzI

! SITE-BASIS
       integer (kind=4), dimension(:,:), allocatable :: basis

! HAMILTONIAN ELEMENTS
       real (kind=8), dimension(:,:), allocatable :: VecI


       end module


!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!             Program starts here
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

       Program IsingModelFixedUp

       use variables
       implicit none


!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! PARAMETERS
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```fortran
      chain=6
      upspins=chain/2
      call SectorDimension()
      dimTotal=facL/facU

      JzI=10.0d0


!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!c CREATE SITE-BASIS
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      allocate(basis(dimTotal,chain))
      call SiteBasis()


!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! ELEMENTS OF THE ISING HAMILTONIAN with FIXED NUMBER OF UP-SPINS
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      allocate(VecI(dimTotal,dimTotal))
      call HamiltonianISING()

      write(*,*)
      write(*,20) 'BASIS', 'DIAGONAL ELEMENTS'
20    Format (3X,A5,4X,A17)
      Do i=1,dimTotal
        write(*,30) (basis(i,j), j=1,chain),  VecI(i,i)
      Enddo
30    Format (6I2,3X,F8.3)

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      deallocate(basis)
      deallocate(VecI)


!ccccccccccccccccccccccccccccccccccccccccccccccccccc
!c END END END END END END END END END END END END
!ccccccccccccccccccccccccccccccccccccccccccccccccccc
!     STOP
      END



!ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!cccccccccccccccccc SUBROUTINES SUBROUTINES SUBROUTINES cccccccccccc
!ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```fortran
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! *************** DIMENSION OF SUBSPACE ***************************
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      subroutine SectorDimension()
      use variables
      implicit none

      facL=1
      Do i=chain-upspins+1,chain
        facL=facL*i
      Enddo

      facU=1
      Do i=2,upspins
        facU=facU*i
      Enddo



!c END of SUBROUTINE for DIMENSION OF SUBSPACE
      return
      end subroutine SectorDimension





!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! *************** WRITING THE SITE-BASIS ***************************
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      subroutine SiteBasis()
      use variables
      implicit none

      INTEGER (kind=4) :: ib,jb

      logical mtc
      integer (kind=4) :: in(chain),m2,h

      mtc=.false.
```

```
!c INITIALIZATION
      Do ib=1,dimTotal
        Do jb=1,chain
          basis(ib,jb)=0
        enddo
      enddo

      ii=1

71   call NEXKSB(chain,upspins,in,mtc,m2,h)
      Do jb=1,upspins
        basis(ii,in(jb))=1
      Enddo
      ii=ii+1
      if(mtc) goto 71

!c END of SUBROUTINE for SITE-BASIS
      return
      end subroutine SiteBasis
!cccccccccccccccccccccccccccccccccccccccccccc
!c SUBROUTINE to get the COMBINATIONS
!cccccccccccccccccccccccccccccccccccccccccccc
      subroutine NEXKSB(n,k,a,mtc,m2,h)
      integer (kind=4) :: n,k,a(n),m2,h,jn
      logical mtc

      if(.not.mtc) then
      m2=0
      h=k
      go to 50
      endif
      if(m2.lt.n-h) h=0
      h=h+1
      m2=a(k+1-h)
   50 do jn=1,h
      a(k+jn-h)=m2+jn
      enddo
      mtc=a(1).ne.n-k+1

      return
      end subroutine nexksb
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!
! SUBROUTINE to write the ISING HAMILTONIAN in the SITE-BASIS
! for a FIXED NUMBER of UP-SPINS
!
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

       subroutine HamiltonianISING()

       use variables
       implicit none

       INTEGER (kind=4) :: tot,bIp(chain)

!c INITIALIZATION
       Do i=1,dimTotal
         Do j=1,dimTotal
           VecI(i,j)=0.0d0
         Enddo
       Enddo

! DIAGONAL ELEMENTS
************************************************************************
       Do i=1,dimTotal
!cccccccccccccc NN cccccccccccccccccccc
         Do j=1,chain-1
       VecI(i,i)=VecI(i,i)+(JzI/4.d0)*(-1.0d0)**(basis(i,j)+basis(i,j+1))
         enddo
!ccccccccccccccccccccccccccccccccccccccc
! CLOSING i=1,dimTotal
       enddo
! END of DIAGONAL
*************************************************************************

!  END of SUBROUTINE that constructs the ISING HAMILTONIAN
!  in the SITE-BASIS for a FIXED NUMBER of UP-SPINS
       return
       end subroutine HamiltonianISING
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```