The first schoolwork of Computational Physics

万炫均 物理 1701 U201710170

Description of this chapter:

This chapter aims to master the basic skill of using Fortran language to program.

1. Input and output

   - Description of the problem:

     Fortran exercise #1: Input and Output

     1) write a fortran program, it can read the following matrix from screen (command window).

     $$\begin{pmatrix} 4 & 2 & 2 & 5 & 8 \\ 2 & 5 & 1 & 3 & 4 \\ 2 & 1 & 6 & 2 & 6 \\ 5 & 3 & 2 & 1 & 3 \\ 8 & 4 & 6 & 3 & 3 \end{pmatrix}$$

     2) output the data to a file named "data.txt".

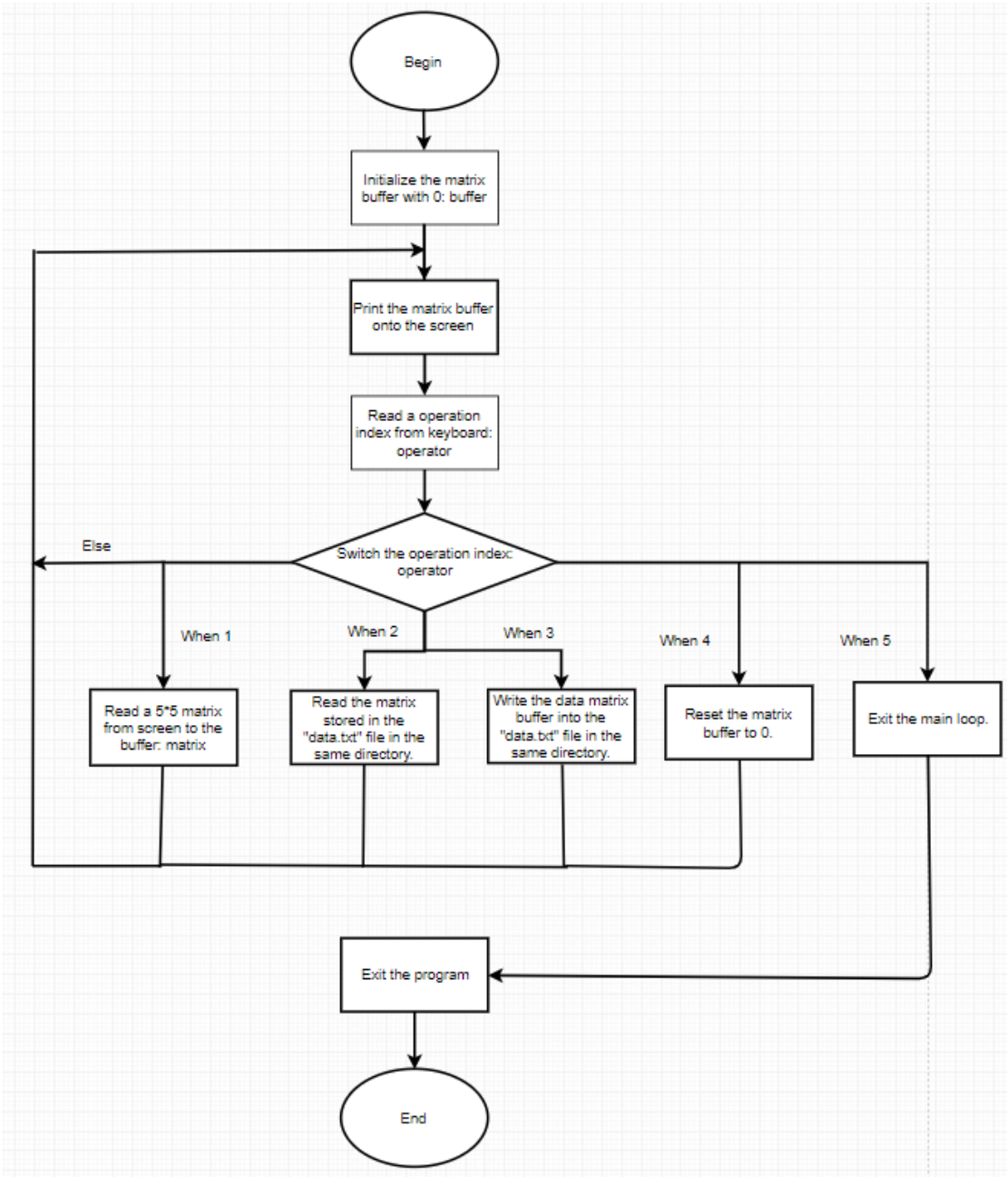
     3) read the data back into the program.

     4) print the data onto the screen (formatting output).

   - The formula to use:

     None

   - Flow chart



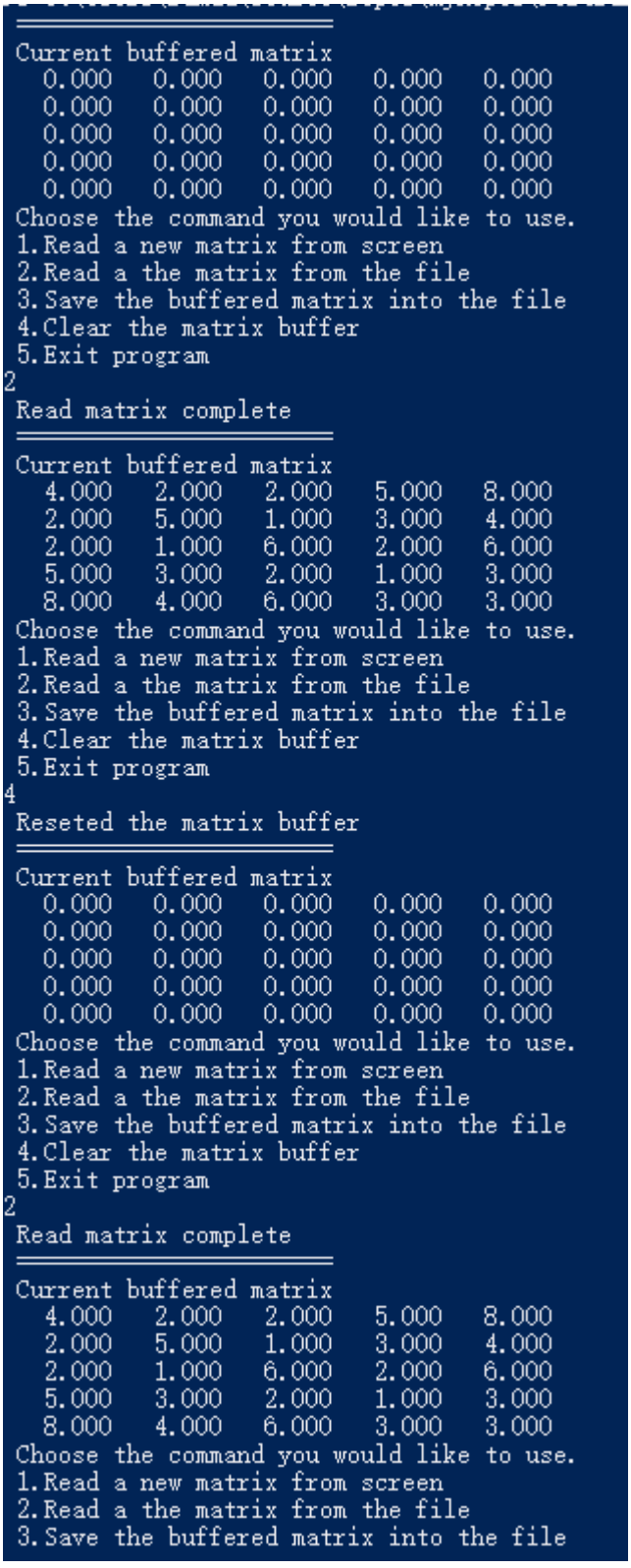
- Source code:

```
program IOProgram
    !Hyper parameter the dim of the matrix
    integer,parameter :: ndim = 5
    real*8 :: matrix(ndim,ndim) = 0
    integer :: operator

    !Main function loop
    do while (.true.)
        print *,"======================"
```

```
        print *,"Current buffered matrix"
        !Print buffered matrix
        do i = 1, ndim
            print"(5f8.3)",matrix(i,:)
        end do

        call ReadOperator(operator)
        select case(operator)
            case (1)
                call ReadMatrixFromScreen(matrix)
            case (2)
                call ReadMatrixFromFile(matrix)
            case (3)
                call SaveMatrixToFile(matrix)
            case (4)
                matrix = 0
                 print *,"Reseted the matrix buffer"
            case (5)
                exit
            case default
                print *,"Wrong operation number!"
                cycle
        end select
    end do

end program

subroutine ReadOperator(operator)
    integer,intent(out) :: operator

    print *,"Choose the command you would like to use."
    print *,"1.Read a new matrix from screen"
    print *,"2.Read a the matrix from the file"
    print *,"3.Save the buffered matrix into the file"
    print *,"4.Clear the matrix buffer"
    print *,"5.Exit program"
    !Program pauses here
    read *,operator

end subroutine

subroutine ReadMatrixFromScreen(matrix)
    real*8,intent(out) :: matrix(5,5)
    print *,"Please enter 25 numbers to form a 5*5 matrix"
```

```
    read *,matrix
    matrix = transpose(matrix)
    print *,"Read matrix complete"
end subroutine

subroutine ReadMatrixFromFile(matrix)
    real*8,intent(out) :: matrix(5,5)

    open(file="./data.txt",unit=10)
    read (10,*)matrix
    matrix = transpose(matrix)
    close(unit=10)

    print *,"Read matrix complete"
end subroutine

subroutine SaveMatrixToFile(matrix)
    real*8,intent(in) :: matrix(5,5)

    open(file="./data.txt",unit=10)
    do i = 1,5
        write(10,"(5f8.3)") matrix(i,:)
    end do
    close(unit=10)

    print *,"Save matrix complete"
end subroutine
```

- Result and example:



- Demo:

  Check the folder ”IOProgram” in the directory.

2. Subroutine

- Description of the problem:

Fortran exercise #2: Subroutine

Write a subroutine to do the matrix and vector operation.

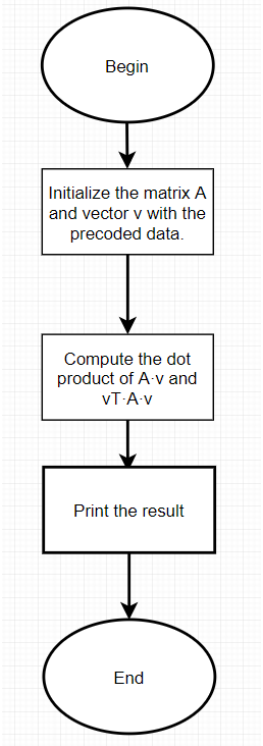
$$A = \begin{pmatrix} 4 & 2 & 2 & 5 & 8 \\ 2 & 5 & 1 & 3 & 4 \\ 2 & 1 & 6 & 2 & 6 \\ 5 & 3 & 2 & 1 & 3 \\ 8 & 4 & 6 & 3 & 3 \end{pmatrix} \qquad b = \begin{pmatrix} 2 \\ 4 \\ 5 \\ 2 \\ 1 \end{pmatrix}$$

Input parameter: matrix A and vector b.
Output parameter: vector A*v and scalar v^T*A*v

- The formula to use:

  The dot product of matrices.

- Flow chart

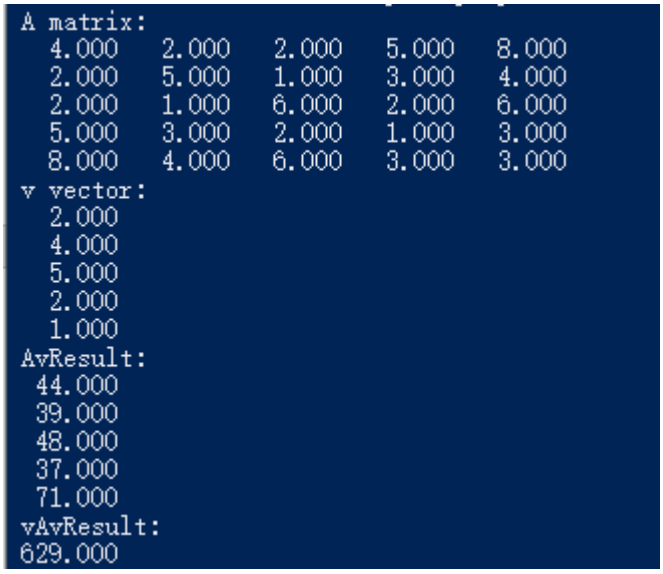


- Source code:

```
program Sub
    implicit none
    integer :: i

    real*8 :: A(5,5)
    real*8 :: v(5,1)

    real*8 :: AvResult(5,1)
    real*8 :: vAvResult(1,1)
    !Initialize
    A = transpose(reshape((/4,2,2,5,8,2,5,1,3,4,2,1,6,2,6,5,3,2,1,3,8,4,6,3,3/),shape(A)))
    v = reshape((/2,4,5,2,1/),shape(v))

    call Calculate(A,v,AvResult,vAvResult)

    !Print the final result
    print *,"A matrix:"
    do i=1,5
        print "(5f8.3)",A(i,:)
    end do

    print *,"v vector:"
    do i=1,5
        print "(f8.3)",v(i,:)
    end do

    print *,"AvResult:"
    do i=1,5
        print "(f8.3)",AvResult(i,:)
    end do

    print *,"vAvResult:"
    print "(f8.3)",vAvResult

end program

!Core calculation routine
subroutine Calculate(A, v, AvResult, vAvResult)
    implicit none
    real*8,intent(in) :: A(5,5)
```

```
    real*8,intent(in) :: v(5,1)
    real*8,intent(out) :: AvResult(5,1)
    real*8,intent(out) :: vAvResult(1,1)

    AvResult = matmul(A,v)
    vAvResult = matmul(transpose(v),matmul(A,v))

end subroutine
```

- Result and example:



- Demo:

  Check the folder ”Subroutine” in the directory.

3. Cycle

- Description of the problem

Write a subroutine to compute the permutation and combination (n=12,m=8)

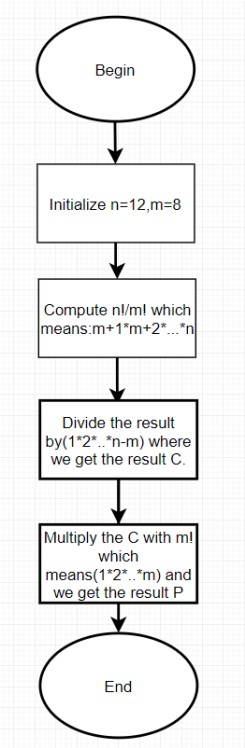
$$P_n^m = \frac{n!}{(n-m)!} \qquad C_n^m = \frac{n!}{m!(n-m)!}$$

Input parameter: n and m
Output parameter: P and C

- Formula to use:

$$P_n^m = \frac{n!}{(n-m)!} \qquad C_n^m = \frac{n!}{m!(n-m)!}$$

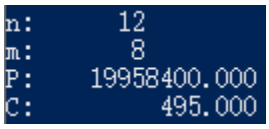
- Flow chart:



- Source code:

```
program CycleProgram
    implicit none
    integer :: n,m
    real*8 :: P,C

    n = 12
    m = 8

    call Calculate(n,m,P,C)

    print "(a,i8)","n:",n
    print "(a,i8)","m:",m
    print "(a,f16.3)","P:",P
    print "(a,f16.3)","C:",C

end program CycleProgram

subroutine Calculate(n,m, P,C)
    implicit none
    integer,intent(in) :: n,m
    real*8,intent(out) :: P,C

    integer :: i

    P = 1
    C = 1
    do i=m+1,n
        C = C*i
    end do
    do i=1,n-m
        C = C/i
    end do
    P = C
    do i=1,m
        P = P*i
    end do
end subroutine Calculate
```

- Result and example:



- Demo:

  Check the folder “Cycle” in the directory.