SDSU | San Diego State University

Lab3 CS420

Eddie Coda
#817061330

Matrix Multiplier

The Prompt: Implement a program that multiplies two matrices in Fortran. Program must have 2 functions: one that multiplies matrices to and another that reads and populates matrices from text files.

Summary

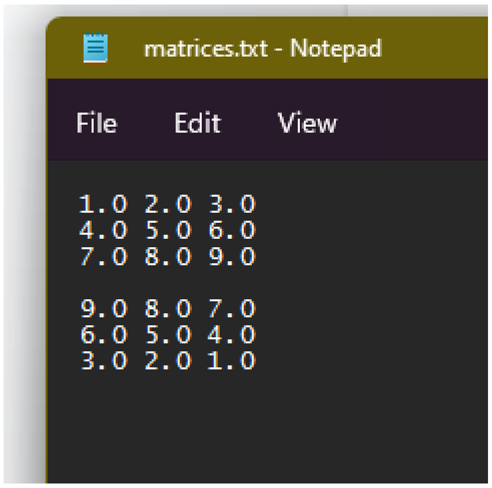
The program assumes that the matrices are stored in a text file called “matrices.txt” in the same directory as the program file “matrixmult001.f95”. The read_matrices subroutine reads the matrices from the file and populates the ‘A’ and ‘B’ arrays. The arrays in the text file must be separated by an extra blank line in between them. The main program multiplies the matrices using a triple nested loop and stores the result in the ‘C’ array. Finally, the program prints the result matrix ‘C’ to the console.

My implementation of this project felt much easier than that of Lab2. I spent way too much time over-analyzing, trying to reconstruct my submission for Lab2 because of the ambiguity in the instructions. However I came to realize that the nature of the vague prompt was not to leave me confused with millions of questions, but instead to give me the freedom to make assumptions at my own will and proceed with what I thought was the best approach. So that is what I did with this project and that is where I will start my report.

The Assumptions:

1. The matrices always have the same size and dimensions: I assumed that both matrices are square matrices of the same size n x n. This means that the program can only calculate the multiplication of matrices that have the same number of rows and columns. I chose this assumption because I wanted to keep the program simple and focused on the task of multiplying matrices. By assuming that both matrices are square, I was able to simplify the program logic, since I only needed to use a single variable ‘n’ to represent the size of both matrices.

2. Input format: I knew that we did not need any special formatting since the prompt only asks to read the input, so I framed the two input matrices as such:



```
1.0 2.0 3.0
4.0 5.0 6.0
7.0 8.0 9.0

9.0 8.0 7.0
6.0 5.0 4.0
3.0 2.0 1.0
```

3. Output format: I knew that the best practice would be to output to a text file, however for initial testing purposes and lack of time towards the end, I simply had the code output the result matrix to the terminal or dos box.
4. The elements of each matrix are separated by spaces.
5. Each row of the matrix is separated by a newline character
6. The number of rows and columns is known and hard coded into the program beforehand.

Overall, these assumptions were made to keep the Fortran program simple and focused on the task of multiplying matrices.

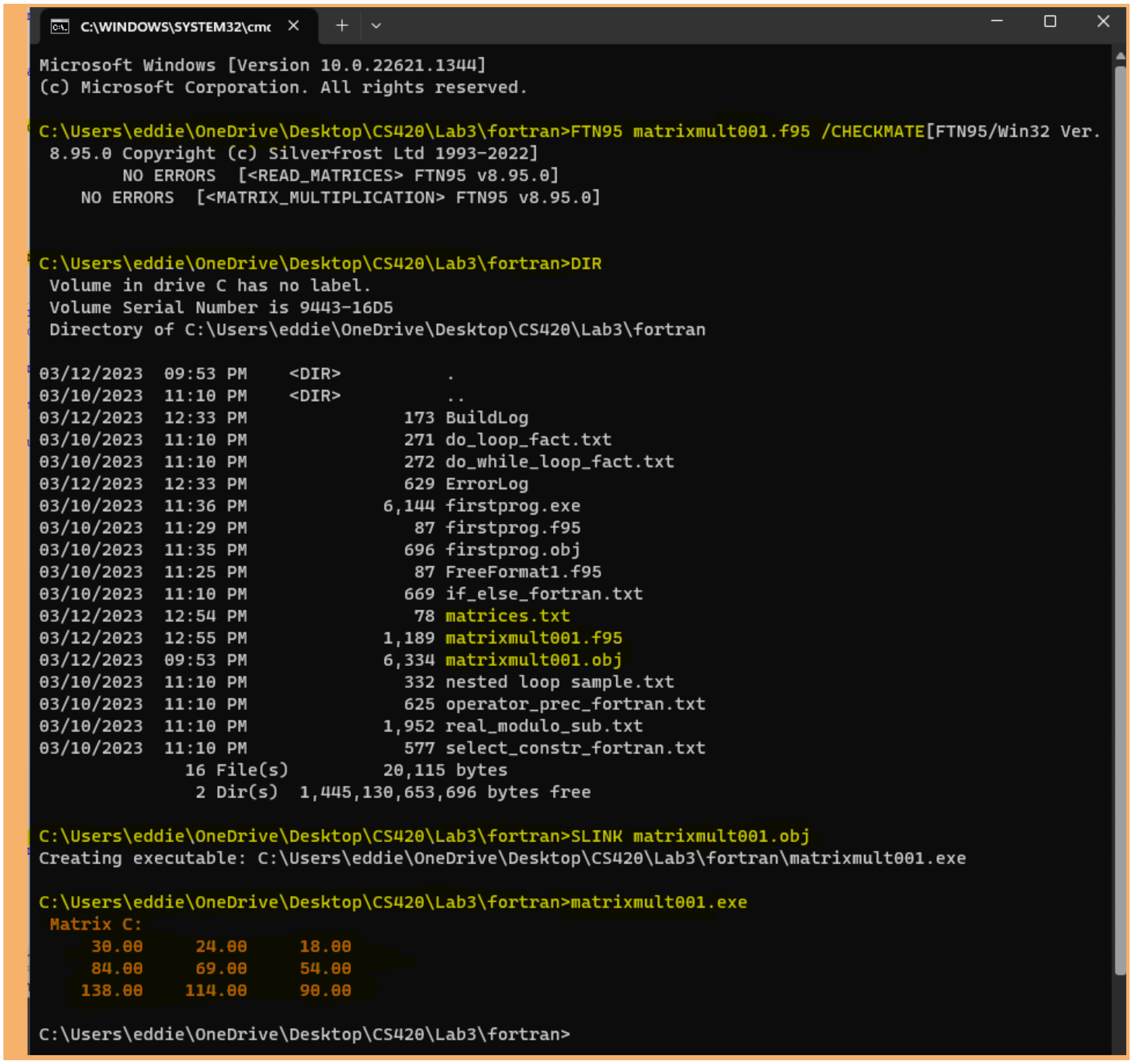
Implementation:

The program defines three matrices ‘A’, ‘B’, and ‘C’, each with dimensions n x n. It also defines three loop indices ‘i’, ‘j’, and ‘k’. The program then multiplies the matrices ‘A’ and ‘B’ together to obtain the result matrix ‘C’, using a nested loop to iterate over the rows and columns of the matrices. The result matrix ‘C’ is printed to the console using the write statement.

The ‘read_matrices’ subroutine takes three arguments: the matrices ‘A’ and ‘B’ to be read from the file, and the size of the matrices ‘n’. The subroutine opens the file ‘matrices.txt’ using the ‘open’ statement, and then uses a loop to read the elements of the matrices from the file using the ‘read’ statement. The subroutine then closes the file using the ‘close’ statement.

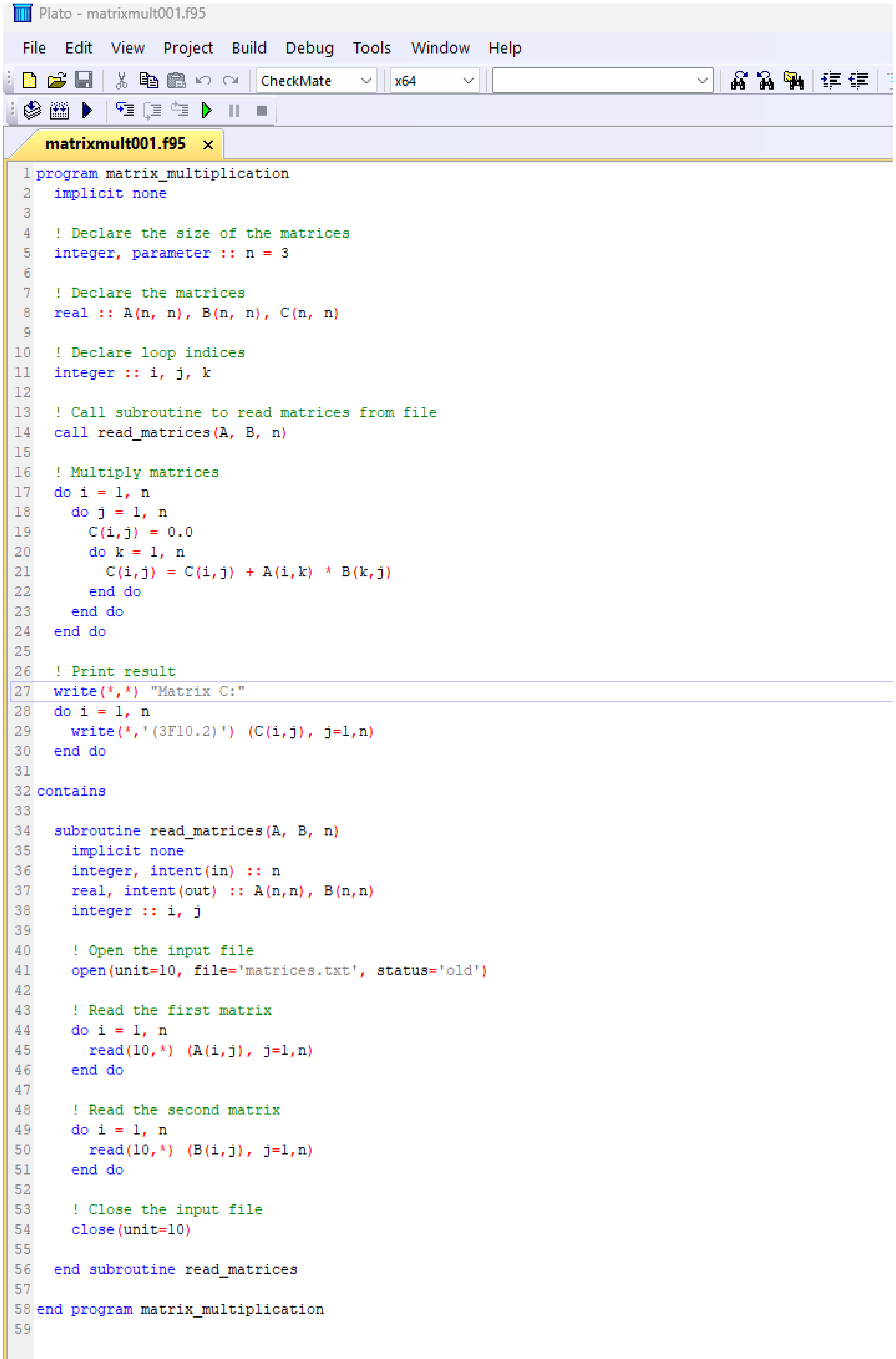
Steps to compile, build and run the code: (text in orange will match the image displayed below demonstrating.)

1. Enter to compile: FTN95 matrixmult001.f95 /CHECKMATE
2. View .obj file was created, Enter: DIR
3. Enter to build: SLINK matrixmult001.obj
4. Enter to run the executable: matrixmult001.exe



```
Microsoft Windows [Version 10.0.22621.1344]
(c) Microsoft Corporation. All rights reserved.

C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran>FTN95 matrixmult001.f95 /CHECKMATE[FTN95/Win32 Ver.
 8.95.0 Copyright (c) Silverfrost Ltd 1993-2022]
        NO ERRORS  [<READ_MATRICES> FTN95 v8.95.0]
    NO ERRORS  [<MATRIX_MULTIPLICATION> FTN95 v8.95.0]


C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran>DIR
 Volume in drive C has no label.
 Volume Serial Number is 9443-16D5
 Directory of C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran

03/12/2023  09:53 PM    <DIR>          .
03/10/2023  11:10 PM    <DIR>          ..
03/12/2023  12:33 PM               173 BuildLog
03/10/2023  11:10 PM               271 do_loop_fact.txt
03/10/2023  11:10 PM               272 do_while_loop_fact.txt
03/12/2023  12:33 PM               629 ErrorLog
03/10/2023  11:36 PM             6,144 firstprog.exe
03/10/2023  11:29 PM                87 firstprog.f95
03/10/2023  11:35 PM               696 firstprog.obj
03/10/2023  11:25 PM                87 FreeFormat1.f95
03/10/2023  11:10 PM               669 if_else_fortran.txt
03/12/2023  12:54 PM                78 matrices.txt
03/12/2023  12:55 PM             1,189 matrixmult001.f95
03/12/2023  09:53 PM             6,334 matrixmult001.obj
03/10/2023  11:10 PM               332 nested loop sample.txt
03/10/2023  11:10 PM               625 operator_prec_fortran.txt
03/10/2023  11:10 PM             1,952 real_modulo_sub.txt
03/10/2023  11:10 PM               577 select_constr_fortran.txt
              16 File(s)         20,115 bytes
               2 Dir(s)  1,445,130,653,696 bytes free

C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran>SLINK matrixmult001.obj
Creating executable: C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran\matrixmult001.exe

C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran>matrixmult001.exe
 Matrix C:
     30.00     24.00     18.00
     84.00     69.00     54.00
    138.00    114.00     90.00

C:\Users\eddie\OneDrive\Desktop\CS420\Lab3\fortran>
```

Sample Code


Plato - matrixmult001.f95
File Edit View Project Build Debug Tools Window Help
CheckMate
x64
matrixmult001.f95

```fortran
program matrix_multiplication
  implicit none

  ! Declare the size of the matrices
  integer, parameter :: n = 3

  ! Declare the matrices
  real :: A(n, n), B(n, n), C(n, n)

  ! Declare loop indices
  integer :: i, j, k

  ! Call subroutine to read matrices from file
  call read_matrices(A, B, n)

  ! Multiply matrices
  do i = 1, n
    do j = 1, n
      C(i,j) = 0.0
      do k = 1, n
        C(i,j) = C(i,j) + A(i,k) * B(k,j)
      end do
    end do
  end do

  ! Print result
  write(*,*) "Matrix C:"
  do i = 1, n
    write(*,'(3F10.2)') (C(i,j), j=1,n)
  end do

contains

  subroutine read_matrices(A, B, n)
    implicit none
    integer, intent(in) :: n
    real, intent(out) :: A(n,n), B(n,n)
    integer :: i, j

    ! Open the input file
    open(unit=10, file='matrices.txt', status='old')

    ! Read the first matrix
    do i = 1, n
      read(10,*) (A(i,j), j=1,n)
    end do

    ! Read the second matrix
    do i = 1, n
      read(10,*) (B(i,j), j=1,n)
    end do

    ! Close the input file
    close(unit=10)

  end subroutine read_matrices

end program matrix_multiplication
```

Source Code Copy:

```
program matrix_multiplication
  implicit none

  ! Declare the size of the matrices
  integer, parameter :: n = 3

  ! Declare the matrices
  real :: A(n, n), B(n, n), C(n, n)

  ! Declare loop indices
  integer :: i, j, k

  ! Call subroutine to read matrices from file
  call read_matrices(A, B, n)

  ! Multiply matrices
  do i = 1, n
    do j = 1, n
      C(i,j) = 0.0
      do k = 1, n
        C(i,j) = C(i,j) + A(i,k) * B(k,j)
      end do
    end do
  end do

  ! Print result
  write(*,*) "Matrix C:"
  do i = 1, n
    write(*,'(3F10.2)') (C(i,j), j=1,n)
  end do

contains

  subroutine read_matrices(A, B, n)
    implicit none
    integer, intent(in) :: n
    real, intent(out) :: A(n,n), B(n,n)
    integer :: i, j

    ! Open the input file
    open(unit=10, file='matrices.txt', status='old')

    ! Read the first matrix
    do i = 1, n
      read(10,*) (A(i,j), j=1,n)
    end do

    ! Read the second matrix
    do i = 1, n
      read(10,*) (B(i,j), j=1,n)
    end do

    ! Close the input file
    close(unit=10)

  end subroutine read_matrices

end program matrix_multiplication
```

Other Documented Notes:

Obvious parts:

- Declaring and initializing variables (e.g. matrices A, B, and C, and the loop indices i, j, and k).
- Using the read statement to read input from a file.
- Using the do statement to loop over the indices of the matrices.
- Performing arithmetic operations (e.g. multiplication, addition).

Not-so-intuitive parts:

- Using subroutines to break up the program into smaller, more manageable pieces.
- Understanding the implicit typing rules in Fortran 95.
- Using the implicit none statement to disable implicit typing and ensure that all variables are explicitly declared.

- Using the parameter keyword to declare constants.
- Understanding the difference between input and output arguments in subroutines, and using the intent keyword to specify argument modes.
- Using the contains statement to define subroutines and functions within a program.
- Using the open and close statements to open and close input and output files.
- Understanding the format of input and output statements, and using the write statement to output results to the console.
- On a scale of 1 to 10, I would rate the complexity of building this program in Fortran 95 as a 4. While there are some not-so-intuitive aspects to the language, such as the implicit typing rules and the use of subroutines, these concepts can be easily learned with some practice and experimentation. Additionally, Fortran 95 has a relatively straightforward syntax, and the program itself is not overly complex. Overall, building this program in Fortran 95 was a good introduction to the language and did not require advanced knowledge or experience.