Pseudocode

Program 1

import libraries

        creating the size variables for my name

        create 2-d arrays that will act was matrices

        fill matrices with respected values

        print all arrays to a file using the printToFile method

printToFile:

method to print a matrix to a specified file name using its num of rows and cols

Program 2

import libraries

        create boolean flags

        create arrays that will have matrices read into them

        create file scanner

        create variables to have user input stores

        prompt the user for input

        while the userInput is not valid loop

            take in user input into strings

            test first string to see if it is a matrix

if string is not a valid matrix then give error

            test second string to see if it is a matrix

            if string is not a valid matrix then give error

        since matrices are valid, find the matrix file the user is looking for using getMatrix

        then load that matrix from the file into the matrices we have ready

check the size of the matrices are the same for addability using method "isAddable" and load that into our boolean flag

If the matrices are addable then add them, print the sum matrix to a file, and tell the user the name of the file

    else give error telling user they are not the same size

isMatrix:

    method to check if user input matches a viable matrix file

isAddable:

    check the size of the matrices match so they can be added

getMatrix

    create matrix to return a matrix from a specified file

printToFile

    method to print a matrix to a specified file name using the base matrices to build the file name

addMatrices

    method to add two matrices

Program 3

import libraries

        create boolean flags

        create arrays that will have matrices read into them

        create file scanner

        create variables to have user input stores

        prompt the user for input

        while the userInput is not valid loop

            take in user input into strings

            test first string to see if it is a matrix

if string is not a valid matrix then give error

            test second string to see if it is a matrix

            if string is not a valid matrix then give error

        since matrices are valid, find the matrix file the user is looking for using getMatrix

        then load that matrix from the file into the matrices we have ready

//check the size of the matrix1 cols is equal to matrix2 rows using method "isMultipliable" and load that into our boolean flag

If the matrices are multipliable then multiply them, print the product matrix to a file, and tell the user the name of the file

else give error telling user they are not the same size

isMatrix:

    method to check if user input matches a viable matrix file

getMatrix

        create matrix to return a matrix from a specified file

printToFile

method to print a matrix to a specified file name using the base matrices to build the file name

isMultipliable

check the size of the matrices match so they can be added

multiplyMatrices

method to multiply two matrices

Program 4

import libraries

create matrices using java library JAMA that will hold addition of matrices and multiplication of matrices

using getMatrix method get the matrices from each file and off of them create matrices using java library JAMA

        only sum and multiply matrices that fit size criteria

        print the first set of matrices to a new file

        next set of matrices: only sum and multiply matrices that fit size criteria

print the next set of matrices to a new file

PrintToFile

method to print a matrix to a specified file name using the base matrices names and type of operation done to build the file name

getMatrix

create matrix to return a matrix from a specified file

Program 5

import libraries

        create boolean flags

        create arrays that will have matrices read into them

        create file scanner

        create variables to have user input stores

        prompt the user for input

        while the userInput is not valid loop

            take in user input into strings

            test first string to see if it is a matrix

            if string is not a valid matrix then give error

            test second string to see if it is a matrix

            if string is not a valid matrix then give error

        since matrices are valid, find the matrix file the user is looking for using getMatrix

        then load that matrix from the file into the matrices we have ready

        take the dot product of the matrixes using the findDotProduct method

        print the product dot product to a file, and tell the user the name of the file

isMatrix

    method to check if user input matches a viable matrix file

getMatrix

    create matrix to return a matrix from a specified file

PrintToFile

    method to print a matrix to a specified file name using the base matrices to build the file name

findDotProduct

method to find the dot product of two matrices

Program 6

import libraries

        create arrays that will have matrices read into them

        since matrices are valid, find the matrix file for each matrix using getMatrix

        then load that matrix from the file into the matrices we have ready

        using the transpose method transpose each matrix into itself

        print each transposed matrix to its own file using its base name

getMatrix

create matrix to return a matrix from a specified file

printToFile

method to print a matrix to a specified file name using the base matrices names to build the file name

Transpose

Method to transpose a matrix

Program 7

import libraries

using getMatrix method get the matrices from each file and off of them create matrices using java library JAMA

        create a list of matrices and a list for their names to dot product each of them respectively

        create matrices to hold dot product and transpose matrices

        iterate through list of vectors taking the dot product of each possible combination (no repeat vectors)

take the transpose of the first matrix so that we can use matrix multiplication since dot product has similar formula

                    take dot product using matrix library multiplication method

                    print the dotproduct to a file using the names of the matrices used in calculation and the type of operation

                    in this case we use 'D' for dot product

take the transpose of each matrix from part 1 and put the transpose back into itself using library method "transpose"

        print each transpose matrix to its own file using its matrix number and 'T' for its operation

printToFile

method to print a matrix to a specified file name using the base matrices names and type of operation done to build the file name

getMatrix

create matrix to return a matrix from a specified file