Documentation:

The documentation should describe what your program does, the approaches you took, and the online resources or libraries you used, or other relevant information for this programming assignment.

Basically, I went through each program and wrote what the code is doing in words. The only outside information I used was my own notes and my previous Programming Assignment 2 (PA2). This is different and more detailed than the pseudocode I wrote for each program.

Part 1:

* Main
  + Creates the matrices A, vectors b, and arrays to hold the solution
  + Call getMatrix
  + Call gaussAndBack
* writeToFile
  + This function determines what output to write to a file based on if the result was undetermined, inconsistent, or real
* getMatrix
  + This function reads in the entirety of the input file and then divides the numbers into matrix A and vector b
* gaussAndBack 🡪 gauss elimination and back substitution
  + This function prints the information to the screen if undetermined/inconsistent
  + If not, it computes the solution based on the determinant or rearranging of A and b

Part 2:

* Main
  + Creates two matrices for the two input files
  + Call getMatrix
  + Call calculations
* calculations
  + This uses my code from PA2 to get the eigenvalues
  + Then it places the dominant eigen value and non-dominant eigen value into their corresponding variables
  + Call getEigenVector for both eigen values
  + Print result
  + Call normalize
  + Print result
  + Call lambdaAndR 🡪 begins the chain reaction of figuring out the final answer
* lambdaAndR
  + Arranges the numbers from calculations into 4x4 matrices of bigLambda, bigR, and the transpose of R
  + Call rLambdaRT
* rLambdaRT
  + Multiplies bigR and bigLambda
  + Multiplies (bigR \* bigLambda) and transposeR
  + Call writeToFile
* writeToFile
  + Prints and writes to file the contents of bigLambda, bigR, and rLambdaRT
* getMatrix
  + Reads in the values from the provided files – ignores the third column for input\_1 and input\_2
* getEigenVector
  + Same as the function in my PA2
* normalize
  + Same as the function in my PA2

Part 3:

* Main
  + Create necessary variables to hold all the different points, input from files, solutions, etc.
  + Call getMatrix for all three input files
  + Transfer values from input files into points
  + Call computeArea for all three collections of points
  + Call computeDistance for all three collections of points
  + Call writeToFile
* writeToFile
  + Just prints the area and distance of each file to the new file
* computeDistance
  + If 2d
    - Calculates slope and y intercept
    - Uses those values to get the distance
  + If 3d
    - Uses a [3] array to condense the calculations of all three points
    - Math functions fabs (absolute value), sqrt (square root), and pow (power) to finalize the calculations for the distance
* computeArea 🡪 was originally going to divide based on 2d and 3d, but 3d worked for both
  + Calculate the cross product of the three points and then use that to figure out the area
* getMatrix
  + Place the numbers in the input files into 2x2 or 3x3 matrices depending on how many numbers are recieved