Pseudocode Program 7

Initialize dimensions (first name, last name)

→ go into function to create matrices:

Creating matrix:
- create empty matrix
- for index in range of rows, append the blank matrix
- for index in range of columns, append current iteration of blank matrix
- set start value as the increment

Populate matrix:
- for index in range of rows,
- for index in range of columns,
- iterate through columns

} 1 function

Go into function to write matrices to files:
- has parameters of a file and matrix
- iterate through rows with for loop,
- write contents to file

} 1 function

Call matrix function, give Dimensions, start, and increment

Print matrix made above ↗

Call the File write function, giving the above matrix and file that is desired

} Run 6 times, with different Parameters so different matrices are made

Pseudocode Program 2

Read in all six matrices made in previous program

Use same write file as in Program 1 code

Make function to add matrices: Pass in 2 matrices

create blank matrix;

for index in range of the length of matrix 1

for index in range of the length of matrix 2

for k in range of dimensions

blank matrix index = matrix1 index + matrix 2 index

Run this code for all possible combinations of 2 matrices that are able to be added together

ıa design.

Pseudocode Program 3

Read in all 6 matrixes in the same way as Program 2.

Use same write file method as Program 2.

Create two different multiply matrix functions in order to multiply bigger matrices together and smaller matrices together:

(same style, just different blank matrix)

```
Create blank 6x6 matrix:
for index in range of length of matrix 1
  for index in range of length of matrix 2, index 0,
   for index in range of length of matrix 2,
     blank matrix index = (matrix 1 index) * (matrix 2 index)
print blank matrix index
```

```
Create 4x6 blank matrix:
  Same code here
```

Call functions for all possible combinations. Some combinations will need one function, where some may need the other

Pseudocode Program 4

Make function to calculate dot product without Numpy:

return sum of $(x * x) + (y * y)$

Make write File function

(same as the ones before)

write in vectors, calculate dot product using function above,

↳ write results to separate files using write File function,

Import graphing libraries

(mainly matplotlib)

↳ Graph vectors, save image to a file

Pseudocode Program 5

Make function to subtract vectors

for index in range of length of vector 1 and vector 2

↳ result equals vector 1 index - vector 2 index

Initialize vectors to subtract

↳ call function to subtract

Make Function to add vectors

for index in range of length of vector 1 and vector 2

↳ result equals vector 1 index + vector 2 index

Initialize vectors to add

↳ call function on said vectors

Import graphing libraries

(matplotlib)

↳ Graph resultant vectors from subtraction and addition,

↳ Save image to a file.

Pseudocode Program 6

initialize blank 6x6 matrix

Define function to transpose the six original matrices

↳ rows equal length of matrix

columns equal length of matrix index 0

Initialize blank matrix

for index in range of columns

↳ set to row

for index in range of rows

↳ append matrix index

append row with blank matrix

return blank matrix

Define write file function

↳ Same one that has been used every time

read file;

call transpose function on matrix

Print new matrix

write new matrix to a new file

} will be the same process for all six matrices.