Upon receiving the coding problems, I began reading all the documents several times and considered possible solutions and subproblems. I then decided to begin with the output Averages problem, as I felt quite comfortable with Python and the challenge seemed intriguing. I considered how to calculate the requested statistics and which data I would need. Initially I thought about storing the data into a list and work with it in this data structure. I realized I didn’t need all patient data with space complexity of O(N), I only needed the total values and total number of patients to calculate averages. I wrote down all the subproblems I could find and then rewrote them in the order the problems would then be solved.

1. read from file
2. pick out each word from each line
3. Identify which information is needed
4. find something that specifies needed statistics in file
5. create and iterate all variables in a loop through file
6. calculate 4 variables desired
7. write to a file on desktop with desired variables
8. create main loop to ask for user input and display/save requested data

I considered only calculating the information the user requests, however the statistics requested will never grow exponentially and are not costly to calculate. Therefore, calculating all statistics would not reduce the speed of the program or any future iteration. While implementing the creation of the four statistics variables I realized I would need to round the resulting values. I chose 2 decimal places of precision as this was utilized in the given document. As an error occurred when trying to write a float to a file, a string was needed to avoid errors if a user requested statistics to be saved to a file. I decided functions should be implemented, as there were specific purposes which could be handled by three functions (displaying statistics, saving statistics, calculating statistics). I worked on the user interface with the idea of accepting user data in a main function, utilizing user input in the form of 0’s and 1’s to specify which data the user is interested in. The final problem to solve in the non-GUI version was how to specify where to create the new file. I found the solution was to take the file location using a combination of string slicing, and the rfind method to save the file in the same directory as the input file.

I still needed to implement exception handling and a GUI version, however many problems still existed.

1. pick a GUI
2. basic framework understanding
3. how to display error messages if invalid file location
4. what errors occur in calculating averages
5. how to accept user input
6. when changing windows scope of previous variables is lost

I chose tkinter as I have some experience with the framework previously and upon further research it seemed like an acceptable option. I began watching a lengthy tutorial on 1.5 speed and when implementing my GUI, I would skip around it. Before writing the code for this file I wrote some of the key ideas drawing out my interpretation for what the UI should look like. There should be a window to enter the file location, then the user should specify the statistics they are interested in. Finally, the data is displayed on a third window. I faced several problems in this pursuit, many I believe are a result of having very little experience with the framework. One of the several problems I faced was creating multiple windows rather than just making all functionality on one window. I found that by looping through button functions I could achieve the functionality I desired. However, if this was a more complex project, I would have invested more time in a better solution which would utilize a frame class to traverse windows without creating many global variables.

I believe that this project was challenging, but manageable. I found countless problems, however nothing bogged down the process for any extended time and I am very happy with the solution I was able to find.