studentSemesterAverage:

I start by initializing the file that will store the data:

```
private static final String FILE_PATH = "studentData.txt";
```

I then format all of the elements in the display that I will need. This includes Jlabels to ask for needed data, text fields to collect the needed data, buttons to process and start the program and more Jlabels to display/output the information in the file.

The next step was to code the "Save to File" button.

(inside try statement)

I coded that when the button underwent an action, it would:
-Convert the inputted grades from a string to a double so that they can be calculated
-I then created a variable named "avg" (double data type) that will calculate and store the data of the average of the grades imputed.
-FileWriter writer = new FileWriter(FILE_PATH, true); to tell the program we are writing to the file
-Then write to the file the inputted data, including student name, grade level, semester, grades, average, and finally "----------" to differentiate the students to make it more clear.

```
writer.write("Student Name: " + studentName.getText() + "\n"); //Student name3
                                                    writer.write("Grade Level: " + gradeLevel.getText() + "\n"); //grade level
                                                    writer.write("Semester: " + semesterNumber.getText() + "\n"); //semester number
                                                    writer.write("Grades: " + g1 + ", " + g2 + ", " + g3 + ", " + g4 + "\n"); //grades
                                                    writer.write("Average: " + avg + "\n");//average
                                                    writer.write("----------\n");
```

-The final step in this try statement is to output to the user that the data has successfully been saved to the file: displayLabel.setText("Saved to file!");

Catch Statement (in case there was a problem with saving the input to the file):

-If there is a problem saving the code to the file, the program will output to the display, "Error saving to file."

The next step in the code is to program the "view file contents" button. This code starts when the button undergoes an action (being clicked)

In the try statement:

The first line, BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH));, opens the file and allows the program to read its texts

The next two lines are to storing the data in the file in a variable. The variable “line” stores 1 line of text from the file, and the variable “all” stores all of the data in the file (but starts empty).

Inside a while loop, the condition has “reader” which reads one line at a time in the file until no lines are left. Each time the loop reads a line, it saves that data to the variable “line.” If the line is not null (meaning the file still has more text), the loop adds the line to the variable “all,” which holds all of the values of the file. Once the file reaches the end, readLine() will return null and stop the loop.

reader.close(); let the file know that no more will be read

After the while loop is completed, the variable “all” (containing all of the file components) is outputted to the user.

```
displayLabel.setText("<html>" + all + "</html>");
```

The Catch statement:

If there is an error with finding and collecting the data from the file, it will output, “Error reading file.”