**CS341 Software Engineering**

**Homework 5**

*Requirements Analysis, Data Structures, Testing, Personal Software Process*

You are encouraged to discuss this assignment with others in the class.

**Overview of the Assignment**

1. Write a GUI Java application to read a set of real numbers from an input file.
2. Analyze your needs and implement your own LinkedList class to store these numbers.
3. Calculate and display the mean and standard deviation of the of this set of real numbers.
4. Use **JFileChooser** to allow the user to select the file of their choice in a GUI manner. You can do this by using **WindowBuilder** to add a button. Add code that opens a **JFileChooser** when the button is pressed. Research **JFileChooser** to see how to go about writing this code.
5. Test your application.
6. Monitor your development using the PSP Form.

See the specific requirements on the next page.

**What to turn in**

1. **Construct the Java GUI application described below. Commit the source code, including all classes and ReadMe file, to GitHub. Code must be well documented.**
2. **Submit the Git repository URL for this application to Moodle.**
3. **Submit the following Word documents to Moodle**
* GitHub URL
* Testing summary.
* Completed final summary for your Personal Software Process: **PSP Form**

**Specific Requirements:**

The program should receive a sample of real numbers from a text file and report the mean and standard deviation of the sample. Begin by analyzing the problem and then develop a design plan for your program.

**Program Features**

1. During input, numbers should be stored internally with a linked list.
2. You will need to implement your own linked list.
3. The numbers input into this program will be stored in an external text file - one number per line.
4. After performing the calculations, the program should output the results clearly and concisely in a GUI text component.
5. You must use the Java programming language to develop this program.

**Code Guidelines**

1. Use OOP.
2. Write efficient code. For example, do not unnecessarily traverse the list.
3. Assure the readability of your code.
4. Perform error checking on input. Output should summarize the occurrences of non-valid input. You may do this any way you wish, such as a simple notification in the results output or something more elaborate.

**Testing**

Produce a test summary, indicating how you ensured that the code is correct. You are responsible for resolving all issues. Document your assumptions and decisions wherever requirements are unclear.

At a minimum, test the following conditions:

1. Test for invalid input, such as "3b4"
2. Test for an empty file as input
3. Test an input file with one line only
4. Test with a file containing empty lines in the middle.

**Personal Software Process (PSP Form)**

* Produce a final summary of your Personal Software Process. See the **PSP Form** in Moodle.
* Document the size (number of lines of code) for your entire program, not including the comment lines.
* Record the time spent on design, coding and testing (in minutes).
* List all the defects you encountered and fixed (indicate the type/root cause).
* The final program should have all defects corrected.
* Retain this data for future use.