# Personal Notebook Assignment 2

Jackson Wiebe

3519635

## Problem 1

First I verified the target number of words in the excerpt given with Microsoft word. 342 words in the document.

I choose to go with the argument input since this would make the program simpler. I appears to me this program is more of a helper or utility program than a GUI or user interface type. So we need parse the arguments, I start by checking we only have 1 target argument and that it ends with a “.txt” extension.

Next I create a new scanner object and feed the given file into that. Scanner requires some exceptions so I create the most common exception of “file not found” and a catch all exception. Some other exceptions could be put in for a deeper analysis of the file.

Finally, we read the file with Scanner.next() which returns words one by one. We increment the counter and dump the word. This should work for files of all sizes since the memory requirements are low, the contents of the file are only read one at a time.

## Problem 2

I started this problem with a modification of the first problem. As user input via command line was required I imported the TextIO library and use the getLnString() method to retrieve a line from the user. Next I ran this through the same checks as the first one. Ensure the file is a\*.txt type and is aviable.

Next I modified my reading loop to grab entire lines instead of each word then we print that to the command line. Waiting for textIO.getLn() each time is an easy way to wait for a user <enter> input. The getLn function will also dumb the rest of the input so multiple triggers don’t happen on the input stream.

Finally I printed the total number of lines in the file. For the given file there is 30 lines which matches my program output.

## Problem 3

Using what I built in the previous problems I picked and chose to build the TextFileReader class and required functions. Adding some conditions for the 100 size array was a little tricky to keep in bounds.

Then I wrapped it all in the required TextFileReaderDemo with some checks for the input arguments as I did in the first problem. It was unclear if we still needed a Problem3.java file since the TextFileReaderDemo contains the main loop.

## Problem 4

Since all 3 arrays are the same size we can use a constant to define their size. This will make the math more readable later. Create 3 float arrays of fixed size we define in the constant.

Then we create a new for loop that iterates the same number of times.

* fill the first array with the loop counter value
* fill the second array with the loop counter value squared
* adds the corresponding elements in the first two arrays and puts the result in the corresponding element of the third array
* Lastly, print out the information in the loops as requested.

Casting the counter from int to float was required to store in the array. You could also get around this by using a float as a pointer but that might lead to problems if the pointer was ever set as a fraction. I also added some string formatting to make the output more readable.

## Problem 5

I started by copying the book class from my previous assignment. Then we need to implement a compare function for later use. I wrote a compreTo function compliant with the java.lang.Comparable class. This will be used in the main thread by the compare function. I also added a display() function for easy printing by the main thread.

For the main thread I created an ArrayList of type books and manually added 12 books from the NY best sellers. I also added a Zebra book to ensure the sorting was working properly and two books with identical names to verify the year sort was working properly.

From there we print out the list as input. Then we sort and print again. The Java String compare is cases sensitive so this may need to be tweaked depending on the final use case.