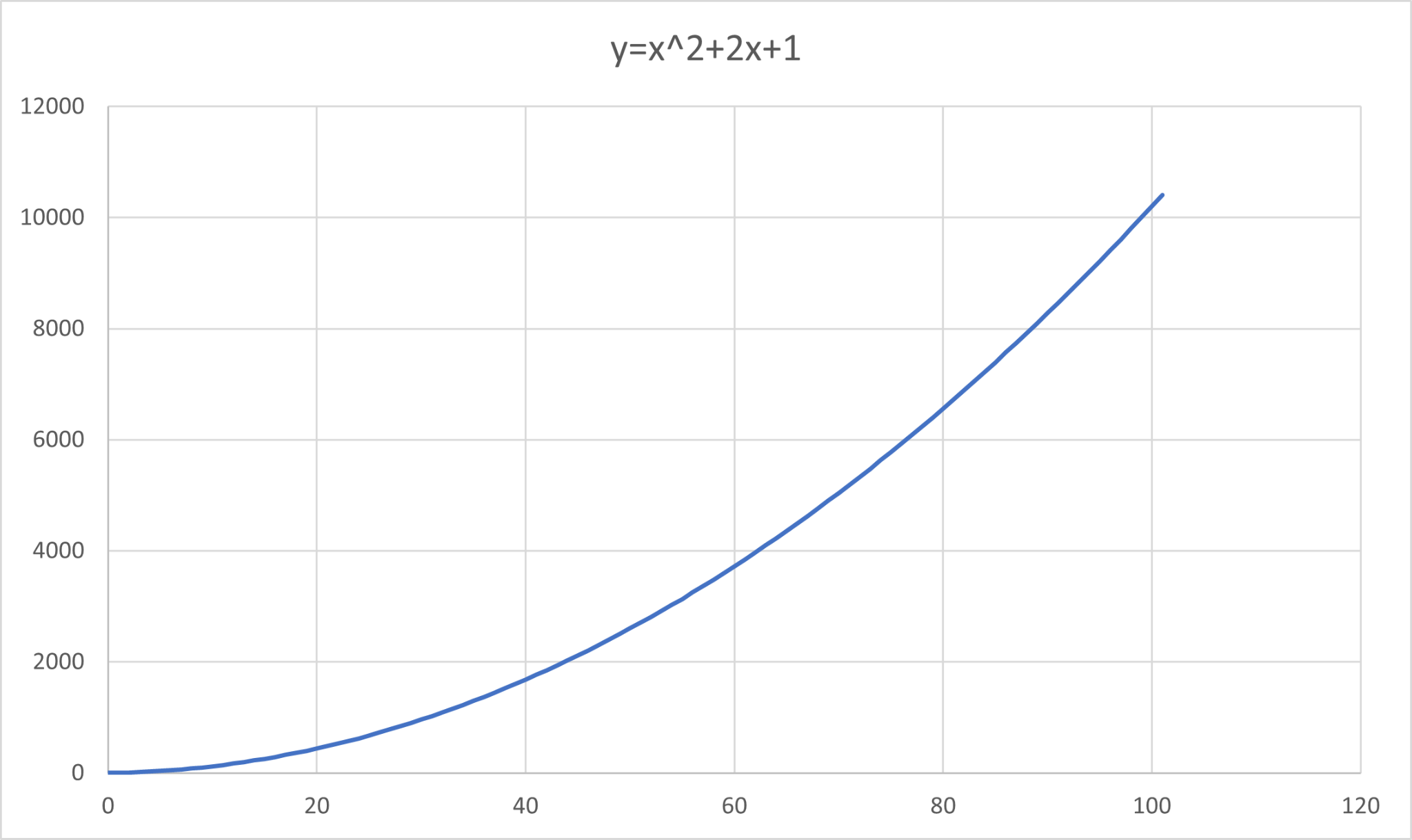
Colin Glory

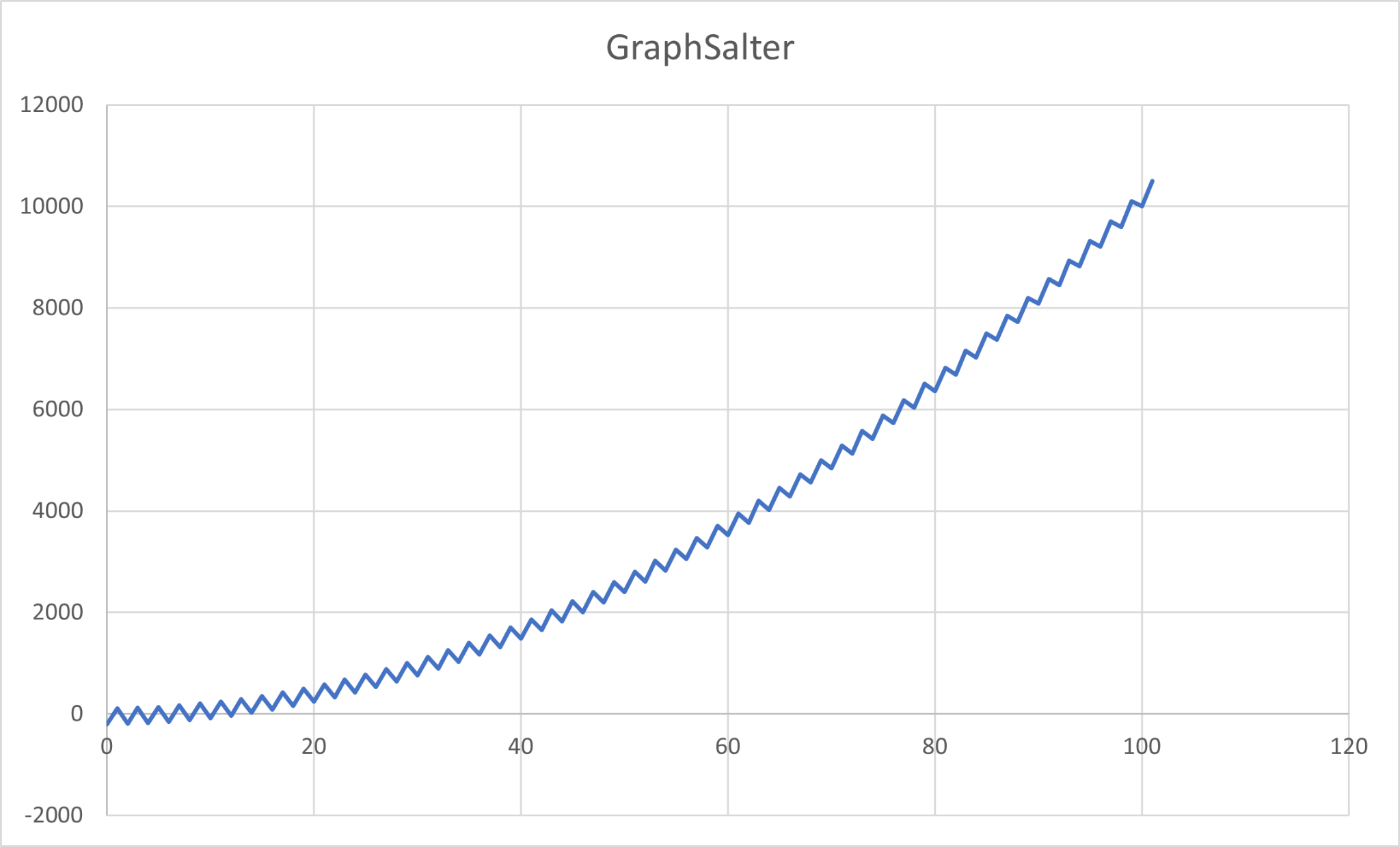
March 25th, 2023

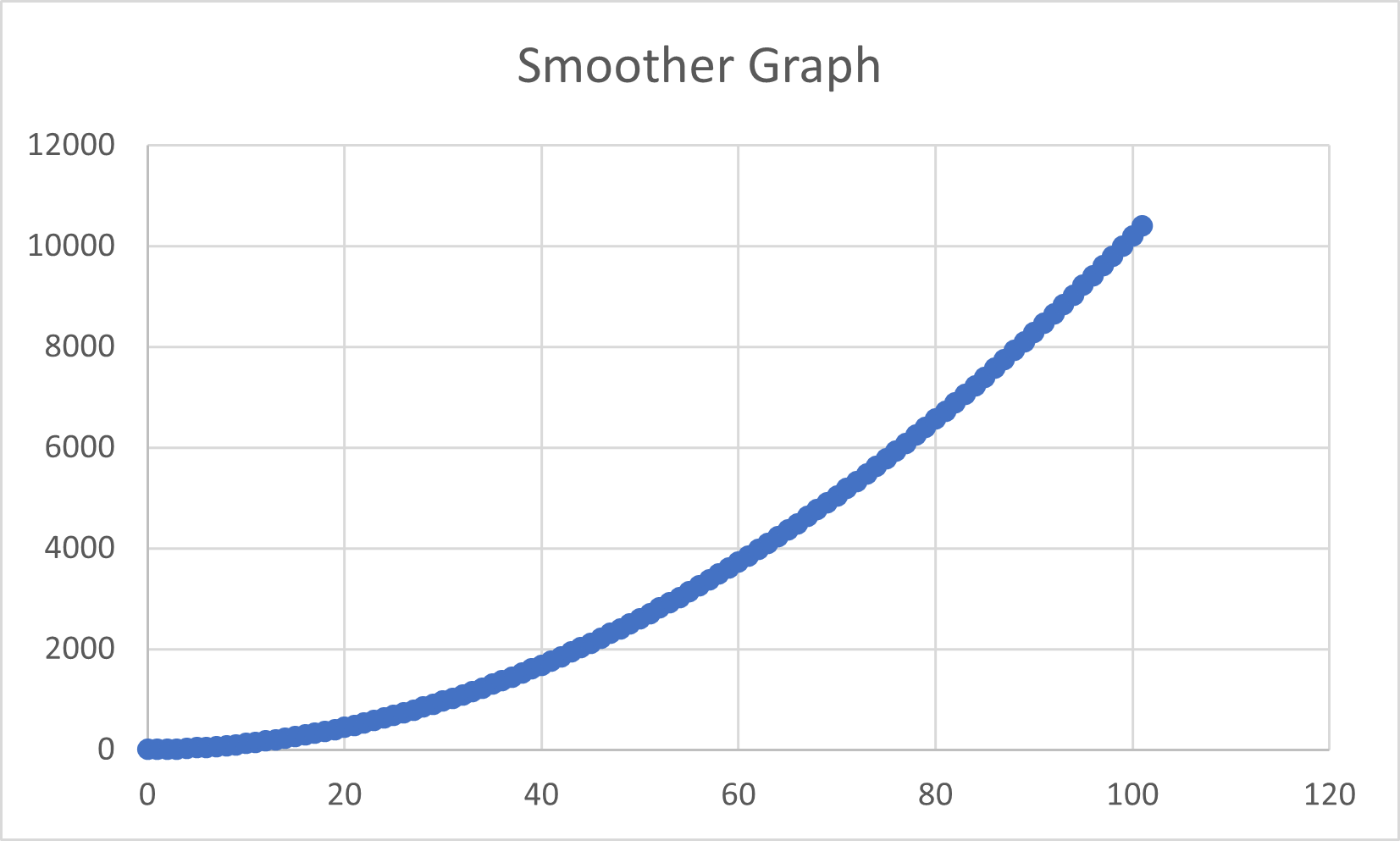
Probability and Applied Statistics

Professor Hoy

CSV Java Graphing Report

In this report, it will dive deep into what I have done to create a csv holding a function’s output, a java salter that will mess up the data/graph, and a smoother that will smooth salted data. The graph formula that I ended up choosing was *y=.* The values 0-100 were iterated through that formula, then written to a csv file. The output of the csv file was categorized with an x column and a y column. The csv file is a file type that is compatible with *Microsoft Excel,* so the file was able to be opened up and edited on Excel. I created a scatter plot with lines to showcase and inspect the x and y values. The graph on the right showcases what you could expect in a sample output for this exercise. By looking at the graph, you could see that as x increases, so does y. This graph does have a smooth curved line throughout all of its values. Moving along, it is time to salt the original csv file.

The term *salting* in this sense, refers to adding/subtracting a random value from your dataset (y values in this case). I did something interesting to shake the salter up a little bit. Rather than just choosing one value, I used two. My salter did not choose a random value, but I decided to go forward with the values 100 and 200. If any of my y values were even, then 100 would be added to the values. If any of the values were odd, then 200 would be subtracted. The original csv file was read into Eclipse, then the x values were parsed to one array and the y values to another. The y value array was put through a for loop to determine whether it was even or add, then the values would be adjusted. The program was finished off with another *FileWriter* where a new csv was created to hold the salted data. 

Finally, it is time to smooth the salted data. I found smoothing the data to be quite easy. It had the same framework as my salter, but obviously the output is smoother. The data was smoothed out due to taking the rolling mean for each value. In other words, I took each value, then added the value in front and the one behind, and finally took the average. The output would become the new number. Smoothing the data will attempt to make the graph and values resemble the original graph/values the more times you smooth it. The picture to the right is the first iteration of the smoother function. Compared to the salter, this obviously resembles the original graph more. I attempted to smooth the data two more times after, each time looking closer and closer to the original graph. 

This programming exercise was a great opportunity to learn about data in Java, as well as learning to manipulate data to look differently. Below I will be attaching all the graphs that I have created.

