In this program I created a FileGrapher program where the program’s goal is to create a csv file and output the plotted values into the csv file. I used the function as my function that I graphed. All the values are already preset. X started at zero, y started at zero, number of iterations are fifty, and the increment value of two. All the values are preset in the loop call. The output graph looks like the following.

Chart, line chart

Description automatically generated

The program for the function is not too crazy. It was simple and was what you expected. As you can see the graph does output the function, so the program is correct.

For the salted program, its job is to randomize the y-values, so the graph doesn’t look like the original graph. I randomized every y-value with a range of 1000. First, I have java find where the java file was in your computer and step back to find the original java file to retrieve the x and y values. I store the values in an array. From there I loop through the array to randomize the y values. To do this I call a method to first see if I am adding or subtracting from the y value. After that I call another method to find a random number between 0 to 999 and either add or subtract that from the y-value. Finally, I take the new y value and replace it with the old one and pass the new x and y values into a csv file to output the graph. The graph looks like the following.

Chart, scatter chart

Description automatically generated

For the smoother the program is a little bit more complex. First, I did the same thing in the salter I had java find where the file was downloaded in the computer and step back in the directory to find the salter csv file. From there I read the file and stored the values in an array. Now I can start smoothing the y values. To start I loop through the array and call the method named averageY. The job of the method is to first check to see where you are in the array. I had a set range of 4 which means at every point it will add the four previous values and the next four values and finally find the average. In the averageY method there are many else if statements. I considered the following cases: (i) the starting point, (ii) second point, (iii) the third point, (iv) the third to last point, (v) the second to last point, and (vi) the last point. The else statement is all the rest of the values. Hence as you can see, depending on what the range will deduce the number of cases there are possible. Once I add up the sum of the y values, I divide that number by the number of values to give me the average value. After the loop was completed, I passed all the values into a csv file to output the graph of the smoother. The graph looks like the following.

Chart

Description automatically generated

After this I wanted to run the same smoother three more times to see what the outputs would look like. So, the following graphs are results from smoothing the previous graphs.

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Lastly you can see that the tail of the graph flattens out. This should not be the case for the graph of . So how can we fix this issue? There are multiple ways to fix this issue. One could just remove the last three elements so the tail of the graph goes away or have the program generate three more values after the last point so there are more values for the tail. I went the easier way of just removing the last three values. This is what the fourth smoother graph looked like.

Chart, scatter chart

Description automatically generated