Arek Ohanian

Prob & Stats

Graphing Project Part1

Part 1 of the project is where I have programmed the graph to write to a csv file. Then to rip the data from that csv file and salt the data and copy it to a new csv file. Lastly to take that data and to smooth it and write that to another csv file. First off, I created an ordered pair class to store the necessary data for later. Which is then used in the function class where it takes in an integer to serve as our x value then sets y to that value plugged into this function . Then the x and resulting y value are put into an ordered pair. This is done this way because in the CSVWritter class which is the class where we write the first CSV file, we have created an array list of ordered pairs, and we fill that list of ordered pairs with x’s from 0 to 99 and store the y as the result from the function above. With help from the DataHandeler class we write the list to a CSV file which can be converted to an excel file which results in this graph, you can change this by changing the math done in the function class if you want to graph a different function.A graph on a graph

Description automatically generated

Next up is salting the data. So first with help from the data handler class which we will go over later we take the data from the original csv file we created earlier and turn it back into an array list of ordered pairs. We then use an import to help with creating a random number, but because it doesn’t create a negative number, we also create another one that is between 0 and 1 and if it is 1 then we make the new number negative. The new number’s range can be changed by altering the bound on randNumb. We then add/subtract from the ripped y value. Where that is then put into a new array list of ordered pairs called saltedPairs. If you make the noise/random number smaller you make the graph closer to the original or if you increase it the result will look less and less like the original. We then use data handler to write the new array list to a csv file which results in this. A graph on a white sheet

Description automatically generated

Next, is the smoother. The CSVSmoother like the salter takes in and rippes a CSV file altough this time we take the data from the salted CSV file. The smoother has a window value that you can edit if you want it to be larger or smaller. With that window value we take the points next to the one we are looking at and average it out with those to get our new value. If it is at the beginning or end of the list then we take it from whatever will add up to the window value times 2. So in the current code the first value takes the 10 in front of it, and reverse for the end. We then write the new y values to the new array list of ordered pairs and write it to a CSV file. The CSV file writing is done with the data handeler class. The result ends up like this.A graph with a line

Description automatically generated

Lastly, there is the data handeler class that I have been mentioning over and over again. Basically because ripping and writing csv’s are done so often we have made a data handeler class to shorten the amount of times I have to write the same code. Data handeler contains two methods one that rips a inputted CSV file and returns the data as an array list of ordered pairs, and the other CSVWritter takes a filename and an array list of ordered pairs and creates a file with the given name in a CSV format. You can change the name of the file by inputting a different file name. You can change the file being read and ripped by inputting a different file. A screen shot of a computer code

Description automatically generated

The main file just calls the methods that will create the CSV files in the order of the original then the salted then the smoother and will create the files and put them into the project 2 folder. A screen shot of a computer program

Description automatically generated