Colin Glory

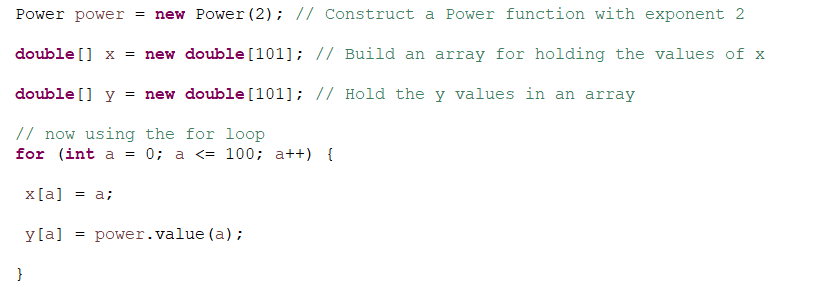
Probability and Applied Statistics

Professor Hoy

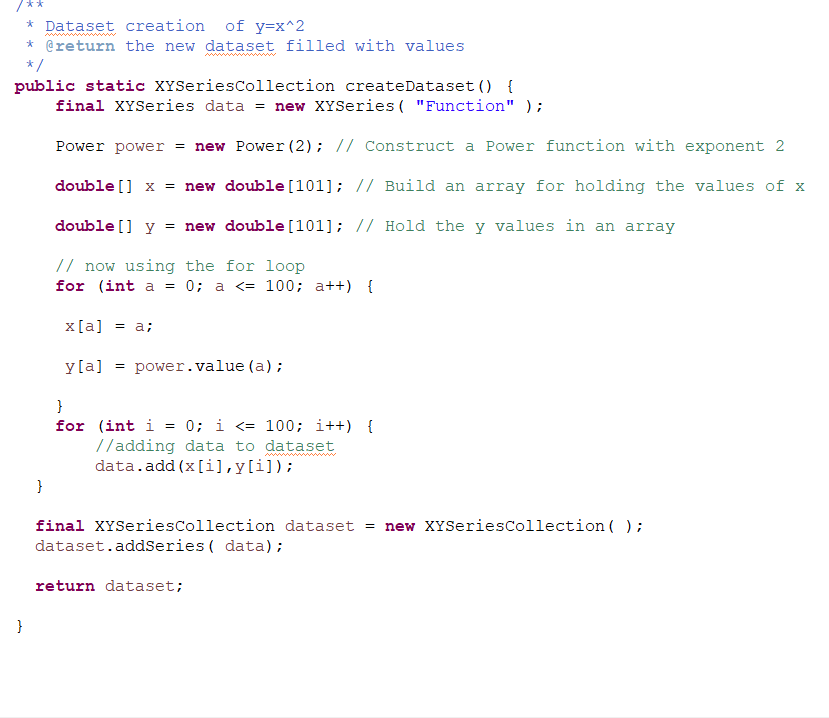
Apache/JfreeChart Report

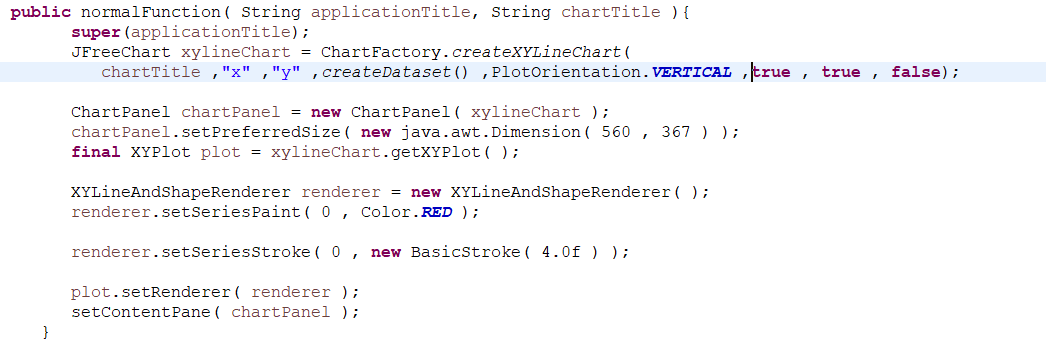
In the previous project, I was tasked with plotting a function, salting it, then smoothing it. All of that work was done using csv readers and writers to create a text file containing the values to being able to open up the csv in Excel. Once in Excel, I had to use the tools to create a graph. Fairly straightforward. This time around, I was tasked with using the *Apache Math Libraries* as well as *JFreeChart*. What is Apache? Well, according to *Apache.org,* “Commons Math is a library of lightweight, self-contained mathematics and statistics components addressing the most common problems not available in the Java programming language or Commons Lang” (Apache 2022). Starting to use Apache at first was quite confusing to understand, but as I worked more with it things became clear.

To use the *Apache* libraries, you will need to go to their site, download section, and choose the most recently updated zip file (will be named something like commons-math4-4.0-beta1-bin.tar.gz). Once downloaded, head over to Eclipse. You may create a new project or edit one to import the libraries. You will want to head over to the project’s build path section, then configure the build path. Classpath must be selected in the libraries section, then select *Add External Jars*. If the downloaded zip file was extracted, then open up the folder and choose the java file, if not extracted then you must extract it. Once it has been added to the classpath, select *apply and close*. Now it is time to use the *Apache Math Libraries*.

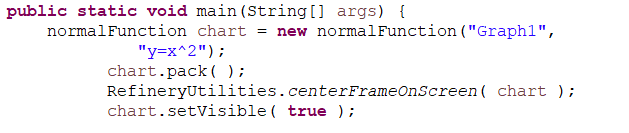
Let’s begin with simply generating data. The function I chose was *y=*. To generate some sample points you may apply this code in a method. The sole difference here is the power function. This function is from the *Apache Libraries* and it allows you to construct exponents fairly easily. These generates points can be written to a csv like the last project, but we can make things more interesting.

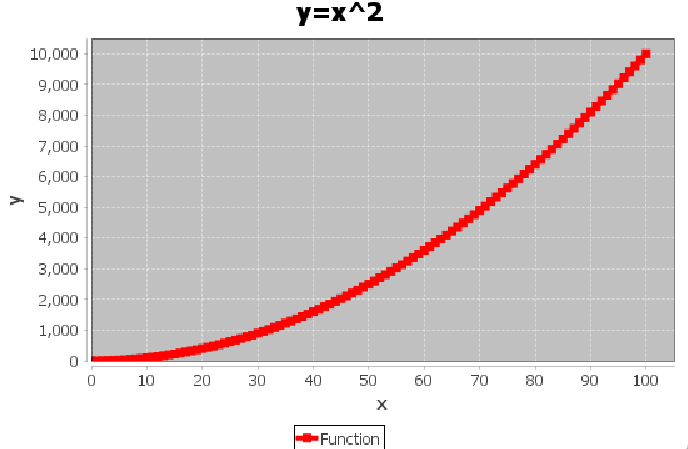
Introducing *JFreeChart.* According to *JavaTPoint.com*, “JFreeChart is a free open source library available for Java which allows users to easily generate professional quality graphs and charts in their applications. One can create pie charts, bar charts (with an optional 3D-effect), line charts, scatter plots, time series charts, Gantt charts, meter charts (dial, compass and thermometer), symbol charts, wind plots, combination charts and more using JFreeChart” (Java 2022). This is simpler then having to create a csv writer method to write the data to a csv, then head to excel to graph the function. You can graph in Eclipse instead. A user would have to follow the same instructions as the *Apache Libraries* to download and install the libraries. The zip file is found on the *JFreeChart* website.

This was the most difficult part of my project. I seemingly wasn't able to understand how to incorporate the library into my code. After messing around with it, I was able to generate multiple plots. After the difficult road, I can say this is more fulfilling than using Excel. I found a tutorial that focuses on graphing using *XYSeriesCollection* data type. This allowed me to construct a series with an empty data set that I would eventually fill. The code on the right shows my method of creating and populating the dataset. I did some more research on how to actually graph the dataset, and eventually was able to come up with the following code:

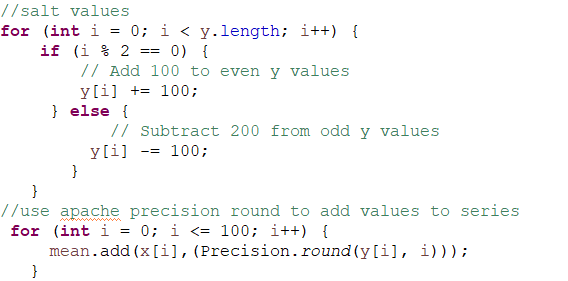


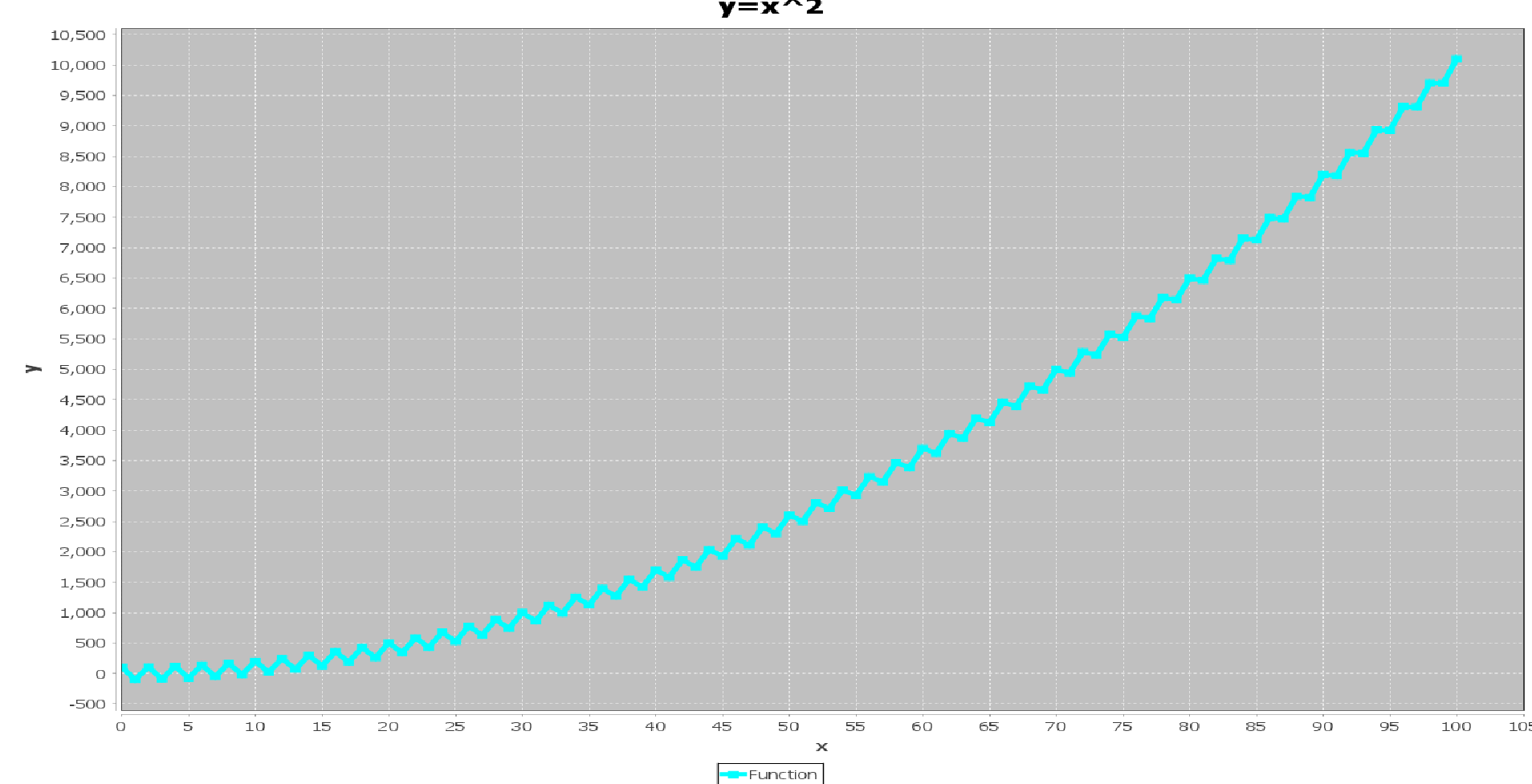
This method allows for the creation of the *JFreeChart* panel, which contains the title, axis’ titles, data set, etc. The size and color of the graph can be adjusted here as well. To run this code, you must have this in your main method:



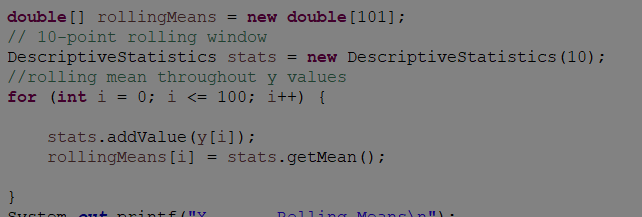
What this does is that the chart is being finalized and then plotted on the screen. Here is a sample output:

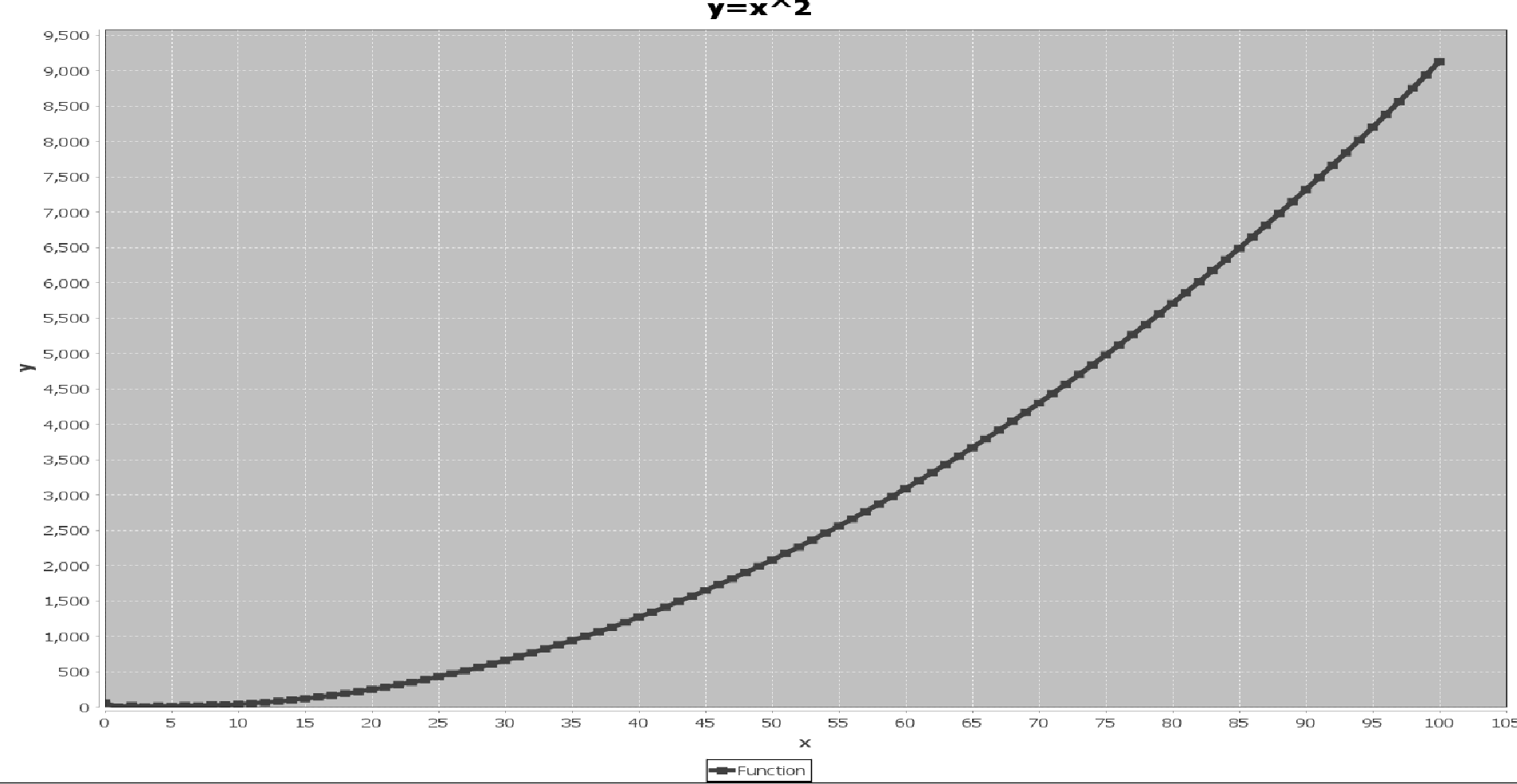
This is a fairly simple process to follow once you are able to wrap your head around it.

Now it is time to salt the data and graph (simply rinsing and repeating). I used the same code as I did in the previous example, but I added in just a couple lines of code to salt the data and add to the dataset. The function *Precision.round* was used from the *Apache* libraries, which essentially rounds the given value to the specified number of decimal places. After slotting in this code, we can now get a graph that looks like this: 



This looks a lot different than the previous graph. I changed the colors here to differentiate between the previous one. By looking at the graph, it is clear that the y values were salted due to the up and down jumps between each point. There is now a distinct difference between the first graph and this one. Now we can move on to smoothing.

Smoothing essentially has the same functionality of salting, but uses a rolling mean to change the y values to give them more of an equivalent look to the original graph/data. The class was the exact same as the salter class, but added one piece of code that goes as follows:

The *Descriptive Stats* instance maintains a dataset of values of a single variable and computes descriptive statistics based on stored data. The windowSize property sets a limit on the number of values that can be stored in the dataset (found in DescriptiveStatistics(10)). All of this allows for a rolling mean to be used to change the salted y values to look more like the untouched data set, which creates a smooth graph. Looking at the graph at the right, you can see that it is no longer salted. This graph more so represents the original data’s graph. The highest point in this graph is around 9,000, while the original graph had a highest value of 10,000. The graph is smooth with no more of a jigged kind of look that the salted graph gives off.

Works Cited

*Apache Commons Math 3.6.1 API. (n.d.). Retrieved April 18, 2023, from https://commons.apache.org/proper/commons-math/javadocs/api-3.6.1/index.html?org%2Fapache%2Fcommons%2Fmath3%2Fstat%2Fdescriptive%2FDescriptiveStatistics.html*

*JFreeChart - XY Chart. Tutorials Point. (n.d.). Retrieved April 18, 2023, from https://www.tutorialspoint.com/jfreechart/jfreechart\_xy\_chart.htm*

*What is Jfree chart - javatpoint*. www.javatpoint.com. (n.d.). Retrieved April 18, 2023, from https://www.javatpoint.com/what-is-jfreechart