In this write-up, I will explain the additions to my code from milestone 1 that allowed me to create a Y-Bus. I have broken it down in sections based on the title of the python files.

**Transformer**

I corrected the arguments in my “Tx” class to eliminate the high and low voltage arguments and to instead have the voltages be passed into the class from the Bus class.

Additionally, I corrected an issue with my code under my “calc\_admittance” function where I used a set zbase instead of calculating the Zpusys based on the power ratings and Sbase.

I added code to build my primitive Y-Bus using pandas for a transformer by first setting the ‘yprim’ variable equal to a 2x2 matrix of zeroes. From there I built a ‘y\_matrix’ function to set the index locations in my ‘yprim’ matrix equal to as calculated previously in milestone 1.

**Transmission Line**

I corrected an issue in my transmission line class where I did not find the per unit values of my transmission line values. I did this by adding a zbase calculation within my ‘calc\_RXB’ function instead of pulling from a set zbase value in the ‘settings’ file.

I added code to build my primitive Y-Bus using pandas for a transmission line by first setting the ‘yprim’ variable equal to a 2x2 matrix of zeroes. From there I built a ‘y\_matrix\_ function to set the index locations in my ‘yprim’ matrix equal to as calculated previously in milestone 1.

**Circuit**

I created a ‘circuit’ class to build functions for the user to input data for the system elements as well as to calculate the overall Y-Bus of the system.

I started by creating a ‘bus\_order’ list, a ‘buses’ dictionary, transformer dictionary, geometry dictionary, conductor dictionary and transmission line dictionary.

The first function I built was the “add\_bus\_element” to allow the user to pass in the name of a Bus as well as the voltage. In this function I utilized an ‘if’ statement to determine if the name of the Bus was already in the dictionary. If it was not in the dictionary, then I appended it to add the name of the Bus.

The second function I built was the “add\_transformer\_element” to allow the user to pass in the name and parameters of the transformer.

The third function I built was the “add\_geomtry\_element” to allow the user to pass in the Distances between the a, b and c phases so my geometry class could calculate Deq.

The fourth function I built was the “add\_conductor\_element” to allow the user to pass in the name and parameters of the conductors to be used in calculating DSL, DSC, reactance and susceptance within the conductor class.

The fifth function I built was the “add\_transmissionline\_element” to allow the user to pass in the name and parameters of the transmission lines.

The final function I built was the “make\_ybus” to calculate the Y-Bus of the system. Within this function, I started by building a matrix of zeroes with the dimensions equal to the number of buses. I then created my Y-Bus by first adding in all the transformer elements based on the indices of each transformer in the system and adding the same indexed elements from the ‘yprim’ matrices together. After adding in all the admittances from the transformers, I duplicated the same process with the transmission lines. In both the transformer and transmission line updates, I created it to update the indexed position within the already created Y-Bus.

**Milestone 2**

My milestone 2 file is my “main” file for this milestone. In here I created the argument of ‘SevenBus’ to be equal to the circuit class from my circuit file. I utilized the add element functions I created in my circuit file to pass in the system parameters for the bus voltages, transformer parameters, transmission line parameters, conductor parameters and the geometry. At the end of this code I ran the “make\_ybus” function from my Circuit to create the Y-bus for the system.