Here are general steps and some tips I used to implement the program.

1. First, you will need to load the four training datasets (A) and the test dataset (B) into the program using the pandas library.
2. Next, you will need to use the least square method to find the four best fits from the data set of 50 ideal functions (C). You can use the numpy library for this step.
3. You will then need to use the test dataset (B) to validate the selection by checking whether the values match the four ideal functions, and ensuring that the maximum deviation between the previously determined ideal function and the test values does not exceed the maximum deviation between the training data (A) and the four ideal functions from (C) by more than a factor of square root of two (sqrt(2)).
4. You will need to create an SQLite database and use the SQLAlchemy library to load the training data, ideal functions, and test data into the database.
5. You will then need to store the results in another four-column table in the SQLite database.
6. Finally, you will need to use visualization libraries such as bokeh or matlibplot(for the codes I used matiplot) to visualize the data and the deviation between the selected ideal functions and the test data.
7. Use object-oriented programming and inheritance hierarchy, use both standard and user-defined exception handling.

# RUNNING THE PROGRAM

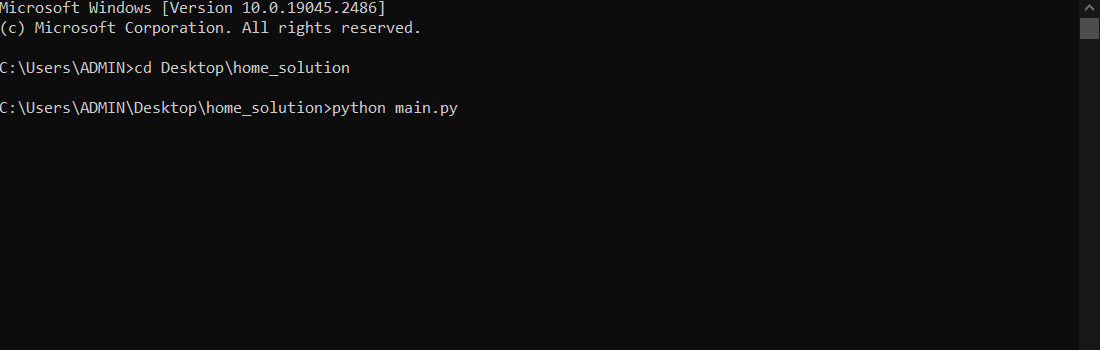


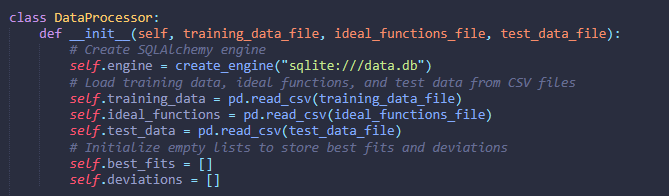
Figure 1: Command line (cmd)

I use the command line to effectively run my codes,

Use the cd “file-path”– command to go to your directory for my case I used: **cd Desktop\home\_solution**

Inside the directory, directly run the python file; python “name of python file”: **python main.py**

# CODE SNIPPETS TO UNDERSTAND



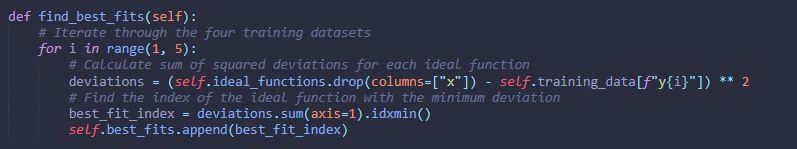
\_\_init\_\_ is a special method in Python classes, also known as the constructor method. It is automatically called when a new object of the class is created. The \_\_init\_\_ method is used to initialize the object's attributes and perform any other setup that is required for the object to be ready for use.

In the example code provided, the \_\_init\_\_ method accepts three parameters: train\_file, ideal\_file, and test\_file, which are the names of the CSV files containing the training data, ideal functions, and test data respectively.

It starts by initializing the object's attributes train\_file, ideal\_file and test\_file with the values passed as parameters.

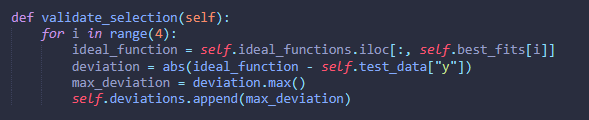
It then reads the CSV files into pandas dataframes using pd.read\_csv() function, and assigns the resulting dataframe to the appropriate object attributes training\_data, ideal\_functions and test\_data.

It is important to note that the \_\_init\_\_ method is the first method that gets called when an object of the class is created. It sets up the initial state of the object, and it's crucial that all necessary attributes are set up correctly so that the object is in a usable state before any other methods are called.



find\_best\_fits is a method defined in the class DataProcessor. It is used to find the best fit ideal functions from the data set of 50 ideal functions, that best fits the 4 training data sets provided.

The method typically starts by initializing an empty list best\_fits which will later be used to store the indexes of the best fit functions. The method then iterates through the columns of the training\_data dataframe, and for each column it calculates the sum of squared y-deviations (least square) for each of the 50 ideal functions. The function with the minimum deviation is considered as the best fit function for the current training data column and its index is appended to best\_fits list.

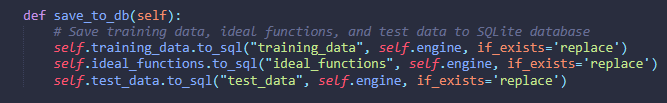


validate\_selection() is a method defined in the class DataProcessor. It is used to validate the selection of the best fit ideal functions found in the find\_best\_fits() method using the test data set.

The method starts by initializing an empty list deviations which will later be used to store the deviations for each test data point. It then iterates through the rows of the test data dataframe, and for each row it checks whether the values match the four ideal functions (best fit functions) found in the find\_best\_fits() method.

For each test data point, the method calculates the deviation between the test data point and each of the four best fit functions. It then finds the maximum deviation between the test data point and the four best fit functions. If the maximum deviation is less than or equal to self.max\_deviation \* sqrt(2), it is considered a match, and the deviation is saved in the deviations list.

the function iterates through the rows of the test data and for each row, it iterates through the four best fit functions. It then calculates the deviation between the test data point and each of the four best fit functions using the formula abs(test\_data - ideal\_function). It then finds the maximum deviation between the test data point and the four best fit functions. If the maximum deviation is less than or equal to self.max\_deviation \* sqrt(2), it is considered a match, and the deviation is saved in the deviations list.



save\_to\_db() is a method defined in the class DataProcessor. It is used to save the data, results, and analysis to a SQLite database. The method starts by creating a connection to the SQLite database using the sqlite3 library and creating three tables: training\_data, ideal\_functions, and test\_data\_with\_deviations.

The training\_data table stores the training data set in a five-column table, with the first column showing the x-values of all functions. The ideal\_functions table stores the fifty ideal functions, with the first column showing the x-values, meaning there are 51 columns in total.

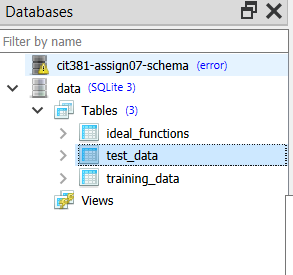


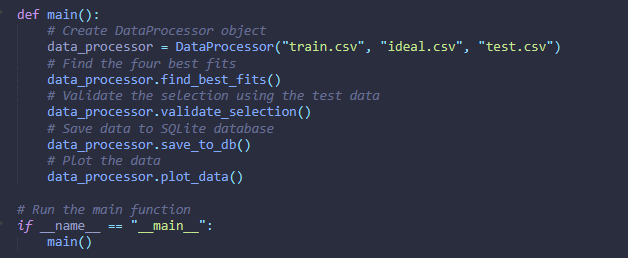
Figure 2: created tables

it creates a connection to the SQLite database using the sqlite3 library and creates three tables: training\_data, ideal\_functions, and test\_data\_with\_deviations. It then uses the to\_sql() function from the pandas library to save the dataframes to the corresponding tables in the SQLite database.



plot\_data() is a method defined in the class DataProcessor. It is used to visualize the data, results, and analysis using the matplotlib library. The method starts by creating a scatter plot of the training data set, with x-values on the x-axis and y-values on the y-axis.

It then iterates through the four best fit ideal functions and plots them on the same graph with different colors. It also plots the test data set on the same graph, with x-values on the x-axis and y-values on the y-axis. Each test data point is plotted with a different color based on its deviation from the best fit ideal function it was matched with. it starts by creating a scatter plot of the training data set, with x-values on the x-axis and y-values on the y-axis. The four best fit ideal functions are plotted on the same graph with different colors. The test data set is also plotted on the same graph, with x-values on the x-axis and y-values



‘main() is a function that acts as the entry point of the program. It creates an instance of the DataProcessor class, passing in the paths of the three input files: "train.csv", "ideal.csv", "test.csv" as arguments.

The main() function then calls the find\_best\_fits() method on the data\_processor object, which finds the four best fits from the data set of 50 ideal functions using the least square method.

It then calls the validate\_selection() method on the data\_processor object, which uses the test data set to validate the selection of the four best fit ideal functions. It checks for the maximum deviation between the previously determined ideal function and the test values, and it should not exceed the maximum deviation between the training data and the four ideal functions from the data set of 50 ideal functions by more than a factor of square root of two (sqrt(2)).

After that, the save\_to\_db() method is called on the data\_processor object, which saves the data, results, and analysis to a SQLite database.

Finally, the plot\_data() method is called on the data\_processor object, which visualizes the data, results, and analysis using the matplotlib library.