1. In this question, you need to create your own synthetic data and do curve fitting on it. First create an input vector x as $[-20, -19, \dots .19, 20]$.

   - Create an output vector $y = a_0 + a_1 x + n$ where a_0, a_1 are some constants that you choose such that $a_0 \neq 0$ and $0.5 \leq a_1 \leq 2.0$. n represents some random noise added to the data. At any data point x_i, you can add some random noise n_i where n_i is a random number between -2 and 2. Here is how you can create a random number between 0 and +1. To create a random number between -2 and +2, you can use `4*(np.random.rand()-0.5)` [Think about why this creates a random number between -2 and +2]. Make sure that the random numbers n_i are all different at different i. Now plot x, y making sure that they are plotted as a scatter (not connected through line segments). Do linear regression on this data to create a model $y = \bar{a}_0 + \bar{a}_1 x$ where $\bar{a}_0, \bar{a}_1$ are the regression estimates of a_0, a_1. How do $\bar{a}_0, \bar{a}_1$ compare with a_0, a_1? Plot the line $y = \bar{a}_0 + \bar{a}_1 x$ on the same plot as the data showing the curve fit model in comparison with the original data.
   - Create an output vector $y = a_0 + a_1 x + +a_2 x^2 + n$ where a_0, a_1, a_2 are some constants that you choose such that $a_0 \neq 0$, $0.5 \leq a_1 \leq 2.0$, and $0.1 \leq a_1 \leq 0.5$. At any data point x_i, you can add some random noise n_i where n_i is a random number between -10 and 10. Plot x, y making sure that they are plotted as a scatter (not connected through line segments). Do quadratic regression on this data to create a model $y = \bar{a}_0 + \bar{a}_1 x + \bar{a}_2 x^2$ where $\bar{a}_0, \bar{a}_1, \bar{a}_2$ are the regression estimates of a_0, a_1, a_2. How do $\bar{a}_0, \bar{a}_1, \bar{a}_2$ compare with a_0, a_1, a_2? Plot the curve $y = \bar{a}_0 + \bar{a}_1 x + \bar{a}_2 x^2$ on the same plot as the data showing the curve fit model in comparison with the original data.
   - Load `data.mat`. In Python you can use loadmat function to load the data file. In Matlab, you can simply do `load('data.mat')`. There are two variables in `data.mat`: x, y. Extract these variables as shown in the above link and fit a polynomial of appropriate degree to this data. What degree works best for a good fit? Plot the fitted curve on the same plot as the data.

2. Download the data from the `stressstrain.csv` data file on Blackboard. `.csv` is an excel format which can be imported in Matlab and Python. In Matlab you can read the data using `T = readtable('stressstrain.csv');` and in Python you can use

   ```
   from numpy import genfromtxt

   T = genfromtxt('stressstrain.csv', delimiter=',')
   ```

   The data comes from an actual tensile test experiment conducted in MMAE419 on stainless steel 316 bar. The data is in the nonlinear regime. The first column is plastic strain values, e, and the second column is corresponding true stress values, s. In the nonlinear regime, the relation between e and s is:

   $$s = K e^{\eta}$$

   where K, η are material constants which need to be determined using curve fitting. See this for more info.

- Plot e, s on a scatter plot
- Find the constants K, η using curve fit

3. In this problem you will do curve fitting on one of the most important datasets on Global Warming, called the Keeling curve dataset. The data and the curve are described here.

   [Charles Keeling is a huge name in atmospheric research and he was associated with the Scripps Institute of Oceanography, which is now part of UCSD. The Keeling curve is very important in our modern understanding of global warming.]

   The data and metadata can be downloaded here but I have also uploaded the data in the file `Keeling-NH.csv`. Read the data in Python or Matlab. You can use the commands provided in `Q2` or here is another way of reading csv files using the powerful library `pandas` (in python):

   ```
   from pandas import read_csv
   dataframe = read_csv('Keeling-NH.csv', header=None)
   data = dataframe.values
   # choose the input and output variables
   x, y = data[:, 3], data[:, 4]
   ```

   x corresponds to year in a decimal format and y corresponds to the measurement of atmospheric CO_2.

   - Plot x, y as a scatter plot. You will notice that there are some negative values of y in the data. Remove these x, y datapoints and create a corrected dataset which we will call x_c, y_c
   - Create a scatter plot of x_c, y_c and limit the y axis limits to between (300,450). This plot shows the increase of atmospheric CO_2 from 1958 to the present day in part per million.
   - Create another scatter plot of x_c, y_c and limit the y axis limits to between (380,420) and the x axis limits to between (2015,2020). This will show you the seasonal variation of atmospheric CO_2 over different years. There is a clear cyclical component to it.
   - The goal is to fit a curve through the data which captures not only the general trend from 1957 to 2021 but also the seasonal variation. Try the following models:

   $$\begin{aligned} y_c &= a_0 + a_1 x_c \\ y_c &= a_0 + a_1 x_c + a_2 x_c^2 \\ y_c &= a_0 + a_1 x_c + a_2 x_c^2 + a_3 \sin(a_4 - x_c) \end{aligned}$$

   - Plot the results of the fitted models on the scatter plot.
   - Based on the fitted models, predict the atmospheric concentrations of CO_2 from 2023 to 2027.