Due data: 9/20/2023, end of the day. **Please submit an .ipynb file via Canvas.**

Instructions:

1) The .ipynb file shall include not only the **source code**, but also necessary **plots/figures** and **discussions** which include your *observations*, *thoughts* and *insights*.
2) Please avoid using a single big block of code for everything then plotting all figures altogether. Instead, use a small bock of code for each sub-task which is followed by its plots and discussions. This will make your homework more readable.
3) Please follow common software engineering practices, e.g., by including sufficient **comments** to functions, important statements, etc.

Programming Problem:

In this problem, you will write a program to estimate the parameters for an unknown polynomial using the polyfit() function of the numpy package.

1) Please plot the noisy data and the polynomial you found (in the same figure). Please use polynomial order of m = 1, 2, 3, 4, 5, 6, 7, 8, respectively.

2) Plot MSE versus order m, for m = 1, 2, 3, 4, 5, 6, 7, 8 respectively. Identify the best choice of m.

3) Change variable *noise_scale* to 200, 300, 400, 600, 800, 1000 respectively, re-run the algorithm and plot the polynomials with the best m found in 2). Discuss the impact of noise scale to the accuracy of the returned parameters. [You need to plot a figure for EACH choice of noise_scale.]

4) Change variable *number_of_samples* to 40, 30, 20, 10 respectively, re-ran the algorithm and plot the polynomials with the best m found in 2). Discuss the impact of the number of samples to the accuracy of the returned parameters. [You need to plot a figure for EACH choice of number_of_samples.]

Please use the following code at the beginning of your program to generate the data.

```
# Simulated data is given as follows in Python:
    import matplotlib.pyplot as plt
    plt.style.use('seaborn-whitegrid')
    import numpy as np
    noise_scale = 100
    number_of_samples = 50
    x =  30*(np.random.rand(number_of_samples, 1) - 0.5)
    y = 2 * x + 11 * x**2 + 3 * x**3 + noise_scale*np.random.randn(number_of_samples, 1)
    plt.plot(x,y,'ro')
```