**Numpy and Matplotlib**

**Problem Statement:**

In this lab, you will investigate the performance difference between a non-vectorized (brute force) and a vectorized implementation of different matrix operations. You will be provided with two functions: ***nonvectorized*** and ***vectorized*** stored in the file ***operation.py***. Each function will accept a single parameter: a 2D NumPy array (an N×N matrix). The ***nonvectorized*** function implements matrix operations in a brute force way.

Your task as follows:

1. Implement vectorized version of operations from ***nonvectorized*** function in ***vectorized*** function. ***vectorized*** function should do similar operations as that of ***nonvectorized***.
2. Import these functions in the file ***main.py.***
3. For a range of N values from 1 to 100 (both inclusive) measure and record the execution time for both functions and store the time values in separate lists for both functions.
4. Plot the execution times against the matrix size N using Matplotlib as a line plot.
5. On the same plot, also draw dashed lines representing the functions N, N², and N³ to serve as references for time complexity. Note: Divide the values of N, N², and N³ by 10^6 for scaling purposes.
6. Based on your plot, estimate the time complexity of each of the two functions. Print these estimates to the terminal as O(N), O(N^2) or O(N^3).

**Usage:** python3 main.py

**Example Output:** Line plot as explained above and on command line:

*Non-vectorized: O(N)*

*Vectorized: O(N)*

(Note: The estimates provided here are for illustrative purposes only)