Practical Session 8
Date: 17/09/2025
Topic: Vectorization and Universal Functions

Solve the following problems using Jupyter Notebook. Please write the following for each of the programming assignments.

1. The problem statement
2. The entire program
3. The sample input
4. The sample output

Please get the program report signed by the instructor.

1. Implement two versions that compute $\sum_i x_i^2$ for 1D array x:
    a. A pure python loop over list[float]
    b. A numpy vectorized version over np.ndarray
    c. Use %timeit to collect timings for sizes n = 1000, 10000, 100000, 1000000 and report it.
2. Construct two counter examples where pure python can match or beat Numpy:
    a. Tiny arrays where function calls overhead dominates
    b. Branching heavy loops where vectorization is impossible.
    c. Provide timings.
3. Create two numpy arrays a and b of size 1 million of type float64.
    a. Preallocate an array res of the same size as a and b using np.empty_like function.
    b. Perform normal addition res = a + b
    c. Use np.add(a, b, out = res) to store results directly.
    d. Compare speed and memory usage and speed with res = a + b.
4. Download *president_heights.csv* which contains the heights of all US presidents. Use the following code snippets to store the heights in an array.

```
import pandas as pd
data = pd.read_csv(path to the file)
heights = np.array(data['heights(cm)'])
print(heights)
```

    a. Calculate the different summary statistics mean, standard deviation, minimum, maximum, median, 25th and 75th percentile.

5. Use np.add (or np.multiply) on a small but illustrative 1-D array:
    a. reduce: sum or product—verify equality with np.sum / np.prod.
    b. accumulate: running sum/product—explain how it differs from reduce.
    c. outer: create an outer sum/product table; connect to broadcasting intuition.