Assignment # 06: Numpy Fundamentals - I

Objective:

To practice array creation, manipulation, and mathematical operations using NumPy — and understand how NumPy enhances performance compared to Python lists.

Part 1: Power of NumPy vs Python Lists

1. Create a list of **10,0000** numbers and a NumPy array of the same numbers.
   - Measure and compare the time taken to **multiply each element by 5**.
   - Which one is faster? Why?

Part 2: Array Creation and Data Types

1. Create:
   - A **1D** array of integers **from 1 to 10**
   - A **2D** array of size **3×3** with numbers from **1–9**
2. Display:
   - **Dimensions, Shape, Size, DataTypes and size of each elements in an array** ---- (for each array created in step 1)
3. Create a **3x3** array of **floats** and convert it into an integer array using **.astype()**.

Part 3: Array Creation Functions

1. Use **NumPy** built-in functions to generate:
   a) A zero matrix of size **4×4**
   b) A one matrix of size **3×2**
   c) An identity matrix of size **5×5**
   d) A constant values array filled with the value **7**
   e) A random integer array of size **(3×4)**, with values between **10 and 99**
2. Use **np.arange()** to generate an array from **5 to 50** with a step size of **5**. Then reshape it into **(3×3)**

Part 4: Indexing and Slicing

Given the array:

```
arr = np.array([[10, 20, 30, 40],
                [50, 60, 70, 80],
                [90, 100, 110, 120]
```

])

1. Extract the **second row**.
2. Extract the **first two rows** and the **second two columns**.
3. Extract the **last column** using slicing.
4. Replace the **middle row** with **[1, 2, 3, 4]**.

Part 4: Vectorization

1. Create two arrays **A** and **B** of **5 random integers each**. Perform element-wise addition, subtraction, multiplication, and division without using loops.

2. Using vectorization, compute the **square root**, **exponential**, and **sine** of all elements in a single array.

[Hint: for computing using np.sqrt(), For computing use np.exp(), For Computing sine using np.sin()]

3. Given a list of **10 temperatures in Celsius**, convert them to **Fahrenheit** using a single vectorized expression.

[Hint: for Celsius to Fahrenheit conversion use formula → °F = (°C × 9/5) + 32]

Part 4: Broadcasting

1. Create a **3×3** matrix – named as **A** and:
   - Add **10** to it (scalar broadcasting).
   - Add a row vector **[1, 2, 3]** to it (row broadcasting).
   - Add a column vector **[[1], [2], [3]]** to it (column broadcasting).
2. Predict the shape of each result before executing the code.

Part 4: Reshape, Flatten, Ravel

1. Create an array of numbers from **1–12**. Reshape it into a **(3×4)** array. Then reshape it again into **(2×6)**.
2. Use **.flatten()** and **.ravel()** on a **2D** array and show the difference between them when you modify the flattened and raveled array.
3. Given an array:

arr = np.arange(1, 10).reshape(3, 3)

   - Convert it into a **1D** array using both methods.
   - Show if changes in one reflect in the original array.