UNIT II

Built-in Data Structures, Functions, Files and Operating System. **NumPy Basics:** Arrays and Vectorized Computation, The Numpy ndarray, Universal Functions, Array-Oriented Programming with Arrays, File Input and Output with Arrays, Linear Algebra, Pseudorandom Number Generation.

**1. Built-in Data Structures in Python**

**Explanation:**

Python provides several built-in data structures, such as:

* **List**: Collection of elements which are Ordered and mutable collections.[ ]
* **Tuple**: Collection of elements which are Ordered and immutable collections. ( )
* **Dictionary**: Collection of pair of elements, Key-value pairs. { }
* **Set**: Collection of elements which are , Unordered collections with no duplicate elements. { }

**Python Example:**

| # List Example fruits = ["apple", "banana", "cherry"] fruits.append("orange") print(fruits) |
| --- |

| # Tuple Example colors = ("red", "green", "blue") print(colors) |
| --- |

| # Dictionary Example person = {"name": "Alice", "age": 25} print(person["name"])   # Set Example numbers = {1, 2, 3, 4, 4} print(numbers) # No duplicate elements |
| --- |

**2. Functions in Python**

**Explanation:**

Functions are reusable blocks of code that perform a specific task. They help organize code and improve modularity.

**Python Example:**

| def greet(name):  return f"Hello, {name}!"   print(greet("Alice")) |
| --- |

**3. Files and Operating System Interactions**

**Explanation:**

Python provides modules like os for interacting with the operating system, and open() for file operations.

**Python Example:**

| # Writing to a file with open("example.txt", "w") as f:  f.write("Hello, World!")   # Reading from a file with open("example.txt", "r") as f:  content = f.read()  print(content)   # OS interaction import os print(os.getcwd()) # Get current working directory |
| --- |

**4. NumPy Basics: Arrays and Vectorized Computation**

**Explanation:**

NumPy (numerical python) provides fast operations on large datasets using arrays. It enables vectorized computations, where operations are applied to entire arrays without the need for loops.

**Python Example:**

| import numpy as np   # Create an array arr = np.array([1, 2, 3, 4]) print(arr \* 2) # Vectorized operation |
| --- |

**5. The NumPy ndarray**

**Explanation:**

ndarray is the core data structure in NumPy, representing a multidimensional array.

**Python Example:**

| arr = np.array([[1, 2, 3], [4, 5, 6]]) print(arr.shape) # (2, 3) print(arr.dtype) # int32 (or int64) |
| --- |

**6. Universal Functions (ufuncs)**

**Explanation:**

Universal functions perform element-wise operations on arrays.

**Python Example:**

| arr = np.array([1, 2, 3, 4]) print(np.sqrt(arr)) # Element-wise square root |
| --- |

**7. Array-Oriented Programming with Arrays**

**Explanation:**

NumPy enables array-oriented programming, making code more concise and faster by avoiding explicit loops.

**Python Example:**

| arr1 = np.array([1, 2, 3]) arr2 = np.array([4, 5, 6]) print(arr1 + arr2) # Element-wise addition |
| --- |

**8. File Input and Output with Arrays**

**Explanation:**

NumPy allows reading and writing arrays to files using functions like np.save and np.load.

**Python Example:**

| # Save an array to a file arr = np.array([1, 2, 3, 4]) np.save("array.npy", arr)   # Load the array from the file loaded\_arr = np.load("array.npy") print(loaded\_arr) |
| --- |

**9. Linear Algebra with NumPy**

**Explanation:**

NumPy provides functions for matrix operations such as dot products, determinants, and eigenvalues.

**Python Example:**

| from numpy.linalg import inv, det   # Matrix operations matrix = np.array([[1, 2], [3, 4]]) print("Inverse:\n", inv(matrix)) print("Determinant:", det(matrix)) |
| --- |

**10. Pseudorandom Number Generation**

**Explanation:**

NumPy provides tools to generate random numbers for simulations and experiments.

**Python Example:**

| # Generate a random array random\_arr = np.random.rand(3, 3) print(random\_arr)   # Generate random integers random\_ints = np.random.randint(1, 10, size=(2, 2)) print(random\_ints) |
| --- |