Python Workshop Outline

1. **Introduction to Python**
   - **What:** A high-level, interpreted programming language.
   - **Why:** Simple, readable, and versatile for research, data science, and AI.
   - **Where:** Used in automation, web apps, simulations, and scientific computing.
   - **When:** Whenever automation or numerical computation is needed.
2. **Programming Basics**
   - What is a program, programming language, and algorithm
   - Typical program structure: comments, variables, operators, control flow, functions
   - Flowcharts and pseudocode for problem-solving
3. **Python Environment Setup**
   - Installing Anaconda, Python, and pip
   - IDEs: Jupyter Notebook, VS Code, Google Colab
   - Running Python scripts and notebooks
4. **First Steps in Python**
   - "Hello, World!" program
   - Input/output: `input()` and `print()`
   - Basic arithmetic and string formatting
5. **Python Standard Libraries**
   - Overview: `math`, `random`, `os`, `sys`, `datetime`
   - Importing modules: `import`, `from ... import`
6. **Data Types**
   - Numbers, Strings, Booleans
   - Type conversion and `type()` function
   - Mutable vs. immutable objects
7. **Collections: Lists, Tuples, Dictionaries, and Sets**
   - **Lists:** indexing, slicing, appending, sorting
   - **Tuples:** immutability, packing/unpacking
   - **Dictionaries:** key-value pairs, iteration, updating
   - **Sets:** union, intersection, difference
8. **Conditional Statements**
   - `if`, `elif`, `else`
   - Logical and comparison operators
   - Nested conditions
9. **Loops**
   - `for`, `while`, and `range()`
   - `break`, `continue`, `pass`
   - Iterating over sequences and dictionaries
10. **Functions**
    - Defining and calling functions
    - Default and keyword arguments
    - Return values and variable scope
11. **Practice Problems (Basic)**

a. Sum of n integers

b. Factorial of a number using a loop

c. Largest of three numbers

d. First 10 Fibonacci numbers

e. Count vowels in a string

f. Reverse a number or string

g. Check if a number is prime

h. Generate a multiplication table

12. NumPy: Numerical Computing

- Array creation: `array`, `zeros`, `linspace`
- Properties: shape, size, dtype
- Indexing, slicing, broadcasting
- Mathematical and statistical operations: `mean`, `std`, `sum`, `exp`, `sin`
- Input/output: saving and loading data

Example Problems:

a. Create a 10×10 random matrix and find max/min

b. Compute dot product of two vectors

c. Generate a sine wave and compute its FFT

d. Simulate dice rolls and compute probability

13. Matplotlib: Data Visualization

- Basic plots: `plot`, `scatter`, `bar`, `hist`, `imshow`
- Labels, titles, legends, and subplots
- Example: plot $\sin(x)$ and $\cos(x)$; histogram of random data

14. Pandas: Data Handling

- Series and DataFrame structures
- Reading/writing CSV and Excel files
- Filtering, grouping, and descriptive statistics
- Plotting directly from Pandas

15. SciPy: Scientific Computing

- Integration (`quad`, `trapz`)
- FFT, Linear Algebra, and Statistics
- Solving equations and ODEs

16. Advanced Problems

a. 1D random walk of 100 steps and print final displacement

b. Exponential decay: $N(t) = N_0 e^{-\lambda t}$ for $t = 0$ to 10

c. Transpose of a 2D matrix

d. Estimate π using Monte Carlo method

e. Numerical integration of $f(x) = x^2$ from 0 to 1 (Trapezoidal rule)

f. Simulate radioactive decay for a set of particles

g. Generate bifurcation diagram for logistic map (optional)

17. Bonus Topics (Optional)

- File handling: `open`, read/write files
- Exception handling: `try`, `except`

- List comprehensions
- Simple animations with Matplotlib

Mini Projects:

- Temperature converter (CF)
- Guess-the-number game
- Dice roll simulator
- Population growth/decay simulation
- Diffusion or sandpile visualization

18. Suggested 3-Day Structure

1. **Day 1:** Basics, Data types, Control structures
2. **Day 2:** NumPy, Matplotlib, Pandas
3. **Day 3:** SciPy and Applied Projects