Python Learning (Beginner to Advanced)

Day 1: Setup & Introduction

- Install Python and your preferred IDE (VS Code, PyCharm, etc.)
- Run your first “Hello, World!” script
- Familiarize yourself with the command line and Python shell

Day 2: Data Types & Variables

- Learn about integers, floats, booleans, and strings
- Practice variable assignment and basic arithmetic operations

Day 3: String Operations

- Explore string concatenation, formatting (f-strings), and slicing
- Work on simple exercises (e.g., reversing a string)

Day 4: Lists & Tuples

- Understand lists (mutable) and tuples (immutable)
- Practice indexing, slicing, and basic list methods

Day 5: Sets & Dictionaries

- Learn about sets (unique items) and dictionaries (key–value pairs)
- Perform simple operations: add/remove elements, access values

Day 6: Basic Operators & Expressions

• Use arithmetic, comparison, and logical operators
• Solve simple math problems and build expressions

Day 7: Control Flow – Conditionals

• Write if, elif, and else statements
• Create simple decision-making programs

Day 8: Loops – For & While

• Master for loops with ranges and iterating over lists
• Practice while loops and understand loop control (break/continue)

Day 9: Functions – Basics

• Define and call functions with parameters and return values
• Understand variable scope and practice with simple examples

Day 10: Mini Project #1

• Build a command-line calculator or a simple text-based game using Days 1–9 topics

---

Day 11: Modules & Standard Library

• Learn to import and use built-in modules (e.g., math, datetime)
• Explore the Python documentation for further learning

Day 12: File Input/Output

• Read from and write to text files
• Practice with CSV files for basic data storage

Day 13: List Comprehensions

• Write concise list comprehensions
• Transform and filter data in lists with single-line expressions

Day 14: Advanced Functions

• Explore lambda functions, map(), filter(), and reduce()
• Practice creating short, anonymous functions

Day 15: Introduction to Object-Oriented Programming (OOP) – Part 1

• Learn how to define classes and create objects
• Understand attributes and methods with simple examples

Day 16: OOP – Part 2

• Dive into inheritance and method overriding
• Create a small class hierarchy (e.g., a simple Animal class and derived classes)

Day 17: Exception Handling

• Learn to use try, except, and finally blocks
• Handle common errors and practice debugging simple code errors

Day 18: Package Management & Virtual Environments

• Install external packages using pip
• Set up and use virtual environments to manage dependencies

Day 19: Working with APIs & JSON

• Learn how to send HTTP requests using the requests module
• Parse JSON data from a public API

Day 20: Web Scraping Basics

• Use requests and BeautifulSoup to scrape data from websites
• Build a small script to extract and display information

Day 21: Decorators

• Understand what decorators are and how they modify function behavior
• Write your own simple decorators to log function calls

Day 22: Generators & Iterators

• Learn how to write generator functions with the yield keyword
• Practice with iterators to handle large data streams efficiently

Day 23: Concurrency – Threads & Multiprocessing

• Get introduced to multithreading and the multiprocessing module
• Write simple concurrent programs to improve performance

Day 24: Asynchronous Programming

• Explore async/await syntax and the asyncio library
• Write a basic asynchronous script for I/O-bound tasks

Day 25: Unit Testing

• Learn the basics of writing tests using unittest or pytest
• Write tests for some of your earlier functions

Day 26: Debugging & Profiling

• Use tools like pdb to step through your code
• Practice profiling your scripts to find performance bottlenecks

Day 27: GUI Programming with Tkinter

• Build a basic GUI application (e.g., a simple form or calculator)
• Understand event-driven programming concepts

Day 28: Web Development Basics with Flask

• Set up a simple Flask application
• Create basic routes and render templates

Day 29: Data Analysis with NumPy & Pandas

• Learn the basics of NumPy for numerical operations
• Use Pandas for data manipulation and analysis on sample datasets

Day 30: Capstone Project

• Integrate multiple topics (file I/O, web scraping, APIs, OOP, web development, or data analysis)
• Build a small real-world application (for example, a Flask dashboard that shows data pulled from an API)
• Document and share your project for feedback