6-Month Python Learning Roadmap

This roadmap is designed to guide you from a beginner to an expert in Python over the course of 26 weeks. It is broken down into five phases, each focusing on a different level of expertise.

Phase 1: Fundamentals (Weeks 1-3)

Weeks 1-2: Basics of Python

- **Introduction to Python:** Understand what Python is and its common uses.
- **Setup:** Install Python and configure your development environment (e.g., VS Code).
- **Core Concepts:** Learn about Python's syntax, variables, data types (e.g., integers, strings, booleans), and basic operations.
- **Control Flow:** Master conditional statements (if, else, elif) and loops (for, while).
- **Modularity:** Learn how to define and call functions, and understand the concept of modules.
- **Project:** Create a simple command-line calculator.

Week 3: Data Structures

- **Built-in Data Structures:** Explore the core data structures:
    - **Lists:** Ordered, mutable collections.
    - **Tuples:** Ordered, immutable collections.
    - **Dictionaries:** Unordered, mutable key-value pairs.
    - **Sets:** Unordered collections of unique elements.
- **Advanced Comprehensions:** Learn to write concise code with list and dictionary comprehensions.
- **Project:** Build a simple contact book application that uses a dictionary to store contact information.

Phase 2: Intermediate Topics (Weeks 4-8)

Week 4: Strings and File Handling

- **String Manipulation:** Master various methods for working with strings (slicing, searching, formatting).
- **Regular Expressions:** Use re module for pattern matching.
- **File I/O:** Learn to read from and write to files.
- **Project:** Develop a text analyzer that counts words, characters, and specific phrases in a file.

Weeks 5-6: Advanced Functions and Modules

- **Advanced Functions:**
  - **Lambda Functions:** Create small, anonymous functions.
  - **Higher-Order Functions:** Use map, filter, and reduce.
  - **Decorators:** Modify the behavior of functions.
  - **Generators:** Create iterators for memory-efficient processing.
- **Project:** Create a URL shortener using functions and modules.

Week 7: Error Handling and Exceptions

- **Exception Handling:** Learn to use try, except, and finally blocks to handle errors gracefully.
- **Custom Exceptions:** Define your own exceptions for specific error conditions.
- **Project:** Build a simple command-line tool that includes robust error handling for invalid input.

Week 8: Object-Oriented Programming (OOP)

- **Core Principles:** Understand classes and objects.
- **OOP Concepts:**
    - **Inheritance:** Create new classes based on existing ones.
    - **Polymorphism:** Use a single interface for different data types.
    - **Encapsulation:** Bundle data and methods that operate on the data.
- **Project:** Build a bank account management system with classes for Account, Customer, etc.

Phase 3: Advanced Python (Weeks 9-14)

Weeks 9-10: Modules and Packages

- **Standard Library:** Explore and use key modules from the Python Standard Library.
- **Custom Modules:** Create your own reusable modules.
- **Package Management:** Learn to install and manage packages using pip.
- **Project:** Create a package for advanced string operations (e.g., tokenization, sanitization).

Weeks 11-12: Working with Data

- **NumPy:** Learn the fundamentals of NumPy for numerical operations.
- **Pandas:** Get an introduction to Pandas and data structures like Series and DataFrames.
- **Data Manipulation:** Practice filtering, sorting, and aggregating data with Pandas.
- **Project:** Perform a simple data analysis on a public dataset (e.g., from Kaggle).

Weeks 13-14: Web Development

- **Introduction to Frameworks:** Learn the basics of a web framework like Flask or Django.

- **Server Setup:** Set up a simple web server and handle requests.
- **Routing and Templates:** Understand how to define URL routes and render HTML templates.
- **Project:** Develop a personal blog website using your chosen framework.

Phase 4: Specialized Topics (Weeks 15-20)

Weeks 15-16: Databases

- **SQLite:** Learn how to work with the built-in SQLite database.
- **SQLAlchemy:** Get an introduction to SQLAlchemy for ORM (Object Relational Mapping).
- **CRUD:** Master Create, Read, Update, and Delete operations.
- **Project:** Build a To-Do application that stores tasks in a database.

Weeks 17-18: Testing and Debugging

- **Unit Testing:** Learn the principles of writing tests for your code.
- **Testing Frameworks:** Use unittest and pytest to write automated tests.
- **Debugging:** Practice debugging techniques using an IDE and print statements.
- **Project:** Write comprehensive tests for one of your previous projects.

Weeks 19-20: Concurrent Programming

- **Multithreading:** Understand how to use threads for concurrent execution.
- **Multiprocessing:** Learn to use multiple processes for true parallelism.
- **Asyncio:** Explore asynchronous programming for I/O-bound tasks.
- **Project:** Develop a simple web scraper that uses concurrent requests to fetch data from multiple websites.

Phase 5: Expert Level Topics (Weeks 21-26)

Weeks 21-22: Advanced Web Development

- **RESTful APIs:** Design and implement a REST API.
- **Authentication:** Add user authentication and authorization to your API.
- **Deployment:** Learn how to containerize and deploy your application using Docker.
- **Project:** Create a REST API for an e-commerce platform.

Weeks 23-24: Data Science and Machine Learning

- **Scikit-Learn:** Get an introduction to the Scikit-Learn library.
- **ML Models:** Build basic machine learning models (e.g., linear regression, classification).
- **Visualization:** Use Matplotlib and Seaborn to visualize data.
- **Project:** Build a simple machine learning model to predict house prices based on a dataset.

Weeks 25-26: DevOps and Cloud Computing

- **CI/CD:** Understand the concepts of Continuous Integration and Continuous Deployment.
- **Automation:** Use tools like Jenkins for build automation.
- **Cloud Deployment:** Deploy a Python application on a cloud platform like AWS or GCP.
- **Project:** Automate the deployment of one of your Python web apps using a CI/CD pipeline.

Continuous Learning and Practice

- **Coding Challenges:** Regularly practice on platforms like LeetCode and HackerRank.
- **Open Source:** Contribute to open-source projects to gain real-world experience.
- **Stay Updated:** Follow Python news and blogs to keep up with the latest trends and

updates.

- **Portfolio:** Continuously update your personal portfolio with your new projects.