**Phase 1: Fundamentals**

**Week 1–2: Basics of Python**

* Introduction to Python
* Installing Python and Setting Up the Environment
* Python Syntax, Variables, and Data Types
* Basic Operations and Expressions
* Conditional Statements (if, else, elif)
* Loops (for, while)
* Functions and Modules
* **Project**: Create a simple calculator

**Week 3: Data Structures**

* Lists
* Tuples
* Dictionaries
* Sets
* List Comprehensions
* Dictionary Comprehensions
* **Project**: Build a contact book

**Phase 2: Intermediate Topics**

**Week 4: Strings and File Handling**

* String Manipulation
* Regular Expressions
* File Handling (Reading/Writing Files)
* **Project**: Develop a text analyzer

**Week 5–6: Advanced Functions and Modules**

* Lambda Functions
* Map, Filter, Reduce
* Decorators
* Generators
* **Project**: Create a URL shortener

**Week 7: Error Handling and Exceptions**

* Exception Handling (try, except, finally)
* Custom Exceptions
* **Project**: Develop a simple command-line tool with error handling

**Week 8: Object-Oriented Programming (OOP)**

* Classes and Objects
* Inheritance
* Polymorphism
* Encapsulation
* **Project**: Build a bank account management system

**Phase 3: Advanced Python**

**Week 9–10: Modules and Packages**

* Python Standard Library
* Creating and Using Modules
* Installing and Managing Packages with pip
* **Project**: Create a package for string operations

**Week 11–12: Working with Data**

* Introduction to NumPy
* Introduction to Pandas
* Data Manipulation with Pandas
* **Project**: Perform data analysis on a sample dataset

**Week 13–14: Web Development**

* Introduction to Flask/Django
* Setting up a Simple Web Server
* Routing and Templates
* **Project**: Develop a personal blog website

**Phase 4: Specialized Topics**

**Week 15–16: Databases**

* Working with SQLite
* Introduction to SQLAlchemy
* CRUD Operations
* **Project**: Build a To-Do application with database integration

**Week 17–18: Testing and Debugging**

* Writing Unit Tests
* Using unittest and pytest
* Debugging Techniques
* **Project**: Write tests for previous projects

**Week 19–20: Concurrent Programming**

* Introduction to Threads
* Multiprocessing
* Asyncio
* **Project**: Develop a web scraper with concurrent requests

**Phase 5: Expert Level Topics**

**Week 21–22: Advanced Web Development**

* RESTful APIs
* Authentication and Authorization
* Deployment with Docker
* **Project**: Create a REST API for an e-commerce platform

**Week 23–24: Data Science and Machine Learning**

* Introduction to Scikit-Learn
* Basic Machine Learning Models
* Data Visualization with Matplotlib and Seaborn
* **Project**: Build a machine learning model to predict house prices

**Week 25–26: DevOps and Cloud Computing**

* Introduction to CI/CD
* Using Jenkins for Automation
* Deploying Python Applications on AWS/GCP
* **Project**: Automate deployment of a Python web app using CI/CD

**Continuous Learning and Practice**

* Participate in Coding Competitions (LeetCode, HackerRank)
* Contribute to Open Source Projects
* Keep Up with Latest Python Trends and Updates
* **Project**: Regularly update and enhance your personal portfolio with new projects

**Resources**

* **Books**: “Python Crash Course” by Eric Matthes, “Automate the Boring Stuff with Python” by Al Sweigart
* **Online Courses**: Coursera, Udemy, edX
* **Documentation**: Official Python Documentation, Real Python

Stay consistent with your learning, practice regularly, and keep challenging yourself with new projects. Good luck on your journey to becoming an expert Python developer, Folks!