### **90-Day Python Learning Plan 🐍🚀**

This structured day-by-day learning plan will take you from a complete beginner to a proficient Python developer, ready to build real-world projects.

## **📌 Days 1–15: Foundational Concepts (Basics & Fundamentals)**

#### **Week 1: Python Basics**

✅ **Day 1** – Introduction to Python, Installation, Running Python Scripts  
✅ **Day 2** – Syntax, Variables, Data Types (int, float, string, boolean)  
✅ **Day 3** – Operators (Arithmetic, Comparison, Logical, Assignment, Bitwise)  
✅ **Day 4** – Strings & String Manipulation (Indexing, Slicing, Formatting)  
✅ **Day 5** – Lists, Tuples, and Sets (Operations, Methods, Use Cases)  
✅ **Day 6** – Dictionaries (Keys, Values, Iteration, Nesting)  
✅ **Day 7** – Python I/O (Reading/Writing Input, File Handling Basics)

#### **Week 2: Control Structures & Functions**

✅ **Day 8** – Conditionals (if, elif, else)  
✅ **Day 9** – Loops (for, while, break, continue, pass)  
✅ **Day 10** – Functions (def, parameters, return values, scope, lambda)  
✅ **Day 11** – Recursion and Modular Programming (Importing Modules, Writing Custom Modules)  
✅ **Day 12** – Error Handling (try, except, finally)  
✅ **Day 13** – Debugging Techniques (print(), logging, pdb)  
✅ **Day 14** – Working with Python Packages (pip, venv)  
✅ **Day 15** – **Mini Project** (Basic Calculator or Simple Text-Based Game)

## **📌 Days 16–45: Intermediate Concepts (Data Structures & OOP)**

#### **Week 3: Data Structures**

✅ **Day 16** – Lists vs. Tuples vs. Sets: Use Cases  
✅ **Day 17** – Stack & Queue Implementation using Lists  
✅ **Day 18** – Dictionary Operations (Sorting, Merging, Nested Dicts)  
✅ **Day 19** – List & Dictionary Comprehensions  
✅ **Day 20** – Advanced String Manipulation (Regular Expressions, re Module)  
✅ **Day 21** – **Mini Project** (To-Do List using Lists & Dictionaries)

#### **Week 4: Object-Oriented Programming (OOP)**

✅ **Day 22** – Introduction to Classes & Objects  
✅ **Day 23** – Constructors (\_\_init\_\_) and Class Variables  
✅ **Day 24** – Inheritance & Method Overriding  
✅ **Day 25** – Polymorphism & Encapsulation  
✅ **Day 26** – Dunder (Magic) Methods & Operator Overloading  
✅ **Day 27** – OOP Best Practices  
✅ **Day 28** – **Mini Project** (Bank Account Management System)

#### **Week 5: Advanced File Handling & Error Handling**

✅ **Day 29** – Reading & Writing Files (open(), read(), write(), with statement)  
✅ **Day 30** – CSV & JSON Handling in Python  
✅ **Day 31** – Working with External Libraries (e.g., requests, matplotlib)  
✅ **Day 32** – Exception Handling (try-except blocks, Raising Custom Errors)  
✅ **Day 33** – Debugging Best Practices (logging, pdb)  
✅ **Day 34** – Regular Expressions for Pattern Matching  
✅ **Day 35** – **Mini Project** (Log File Analyzer)

## **📌 Days 46–75: Advanced Concepts & Problem-Solving**

#### **Week 6: Algorithms & Problem-Solving**

✅ **Day 36** – Understanding Time Complexity (Big-O Notation)  
✅ **Day 37** – Sorting Algorithms (Bubble, Selection, Insertion, Merge, Quick)  
✅ **Day 38** – Searching Algorithms (Linear Search, Binary Search)  
✅ **Day 39** – Recursion & Backtracking Problems  
✅ **Day 40** – **Mini Project** (Sorting Visualizer)

#### **Week 7: APIs, Web Scraping & Databases**

✅ **Day 41** – Introduction to APIs (REST, JSON) & Making API Calls  
✅ **Day 42** – Web Scraping with BeautifulSoup & Scrapy  
✅ **Day 43** – Introduction to Databases (SQL Basics, SQLite, PostgreSQL)  
✅ **Day 44** – Working with Python & Databases (sqlite3, SQLAlchemy)  
✅ **Day 45** – **Mini Project** (Web Scraper & Data Storage in a Database)

#### **Week 8: Performance Optimization & Memory Management**

✅ **Day 46** – Understanding Python Memory Model  
✅ **Day 47** – Optimizing Code for Performance (map(), filter(), reduce())  
✅ **Day 48** – Profiling & Benchmarking Python Code  
✅ **Day 49** – Multi-threading & Multi-processing  
✅ **Day 50** – **Mini Project** (Parallel Processing Task Scheduler)

## **📌 Days 76–90: Real-World Project-Based Learning**

#### **Week 9-10: Build 2-3 Real-World Projects**

🔹 **Day 51-55** – Build a **To-Do List App** (GUI with Tkinter or CLI-Based)  
🔹 **Day 56-60** – Create a **Weather App** using an API (requests, JSON)  
🔹 **Day 61-65** – Develop a **Blog Platform** (Flask/Django + Database)  
🔹 **Day 66-70** – Implement a **Simple Game** (Tic-Tac-Toe or Hangman)  
🔹 **Day 71-75** – Refactoring & Documenting Code, Performance Optimization

## **📌 Additional Resources**

### **📚 Recommended Books & Documentation**

* **Python Crash Course** by Eric Matthes
* **Automate the Boring Stuff with Python** by Al Sweigart
* **Python Documentation**: [docs.python.org](https://docs.python.org/3/)

### **💻 Coding Practice Platforms**

* **LeetCode** ([leetcode.com](https://leetcode.com/)) – Problem-solving
* **HackerRank** (hackerrank.com) – Python challenges
* **Codecademy** (codecademy.com) – Interactive learning

## **📌 Evaluation & Feedback**

✅ **Weekly Checkpoints** – Review progress & test understanding with small challenges  
✅ **Code Reviews** – Share your work on **GitHub**, get feedback from peers  
✅ **Community Support** – Join Python forums like **Stack Overflow, Reddit (r/learnpython), Discord coding servers**

## **📌 Tips for Staying Consistent & Motivated**

🎯 **Set Daily Goals** – Small, manageable tasks make learning easier  
🔄 **Practice Every Day** – Even 30-60 minutes daily will show progress  
💡 **Apply What You Learn** – Work on small projects alongside learning  
📢 **Ask for Help** – Join communities, ask questions, and engage with mentors

### **🚀 By the end of 90 days, you’ll have:**

✅ Strong Python fundamentals 🚀  
✅ Hands-on experience with **projects** 🛠️  
✅ Problem-solving skills 💡  
✅ A **portfolio** of real-world Python applications 💻

🔹 **Final Step**: **Apply for internships, contribute to open-source projects, or start freelancing!**

Would you like recommendations on specific projects based on your interests? 😊