PYTHON PROGRAMMING COURSE SYLLABUS

Official Training Portal • Email: codeveins01@gmail.com

MODULE 01: INTRODUCTION TO PYTHON

- What is Python? & History of Python
- Core Features & Real-World Applications
- Environment Setup: Installing Python & VS Code
- Running Programs & Understanding Execution (Interpreter vs Compiler)
- Writing Your First Python Program
- Code Documentation: Comments in Python

MODULE 02: PYTHON BASICS

- Variables & Naming Conventions
- Data Types: Integer, Float, String, Boolean, None
- Type Casting (Explicit & Implicit conversion)
- Handling User Input & Output Formatting
- Python Reserved Keywords
- Operators: Arithmetic, Comparison, Logical, Assignment, Membership, and Identity

MODULE 03: STRINGS & MANIPULATION

- Creating Strings & Memory Representation
- String Indexing & Slicing Mechanics
- Built-in String Methods
- String Formatting: f-Strings & Legacy formats
- Escape Characters
- Practical String Manipulation Programs

MODULE 04: CONDITIONAL STATEMENTS

- Control Flow Structures
- if, if-else, and if-elif-else Statements
- Nested Conditional Logic
- Practical Examples & Mini Projects

MODULE 05: LOOPS & ITERATION

- Definite Iteration: for Loops
- Indefinite Iteration: while Loops
- Nested Loop Architectures
- Loop Control: break, continue, and pass Statements
- Comprehensive Loop Exercises & Pattern Printing Programs

MODULE 06: DATA STRUCTURES: LISTS

- Creating Lists & Accessing Elements
- Modifying & Updating Lists
- Built-in List Methods & List Slicing
- Nested Lists (Multi-dimensional structures)
- Advanced Concept: List Comprehension
- Real-world Implementation Examples

MODULE 07: DATA STRUCTURES: TUPLES

- Creating Immutable Sequences (Tuples)
- Tuple Operations & Built-in Methods
- Data Packing & Unpacking Techniques
- Comparative Analysis: Tuple vs List

MODULE 08: DATA STRUCTURES: SETS

- Creating Unique Collections (Sets) & Set Methods
- Mathematical Set Operations: Union, Intersection, Difference, and Symmetric Difference
- Practical Programs using Sets

MODULE 09: DATA STRUCTURES: DICTIONARIES

- Creating Key-Value Pairs (Dictionaries)
- Accessing Values & Updating Data Safely
- Built-in Dictionary Methods
- Nested Dictionaries
- Advanced Concept: Dictionary Comprehension
- Real-world Industry Use Cases

MODULE 10: FUNCTIONS & FUNCTIONAL PROGRAMMING

- What are Functions? Rules for Creating Functions
- Parameters vs Arguments
- The return Statement Architecture
- Argument Mapping: Default, Keyword, and Variable Length Arguments
- Anonymous Functions: Lambda Functions
- Core Concepts: Recursion & Scope of Variables

MODULE 11: MODULES & PACKAGES

- Modularity Architecture: What are Modules?
- The import Statement & Variants
- Creating Custom Modules
- Structuring Python Packages
- Deep Dive into Essential Built-in Modules: math, random, datetime, and os

MODULE 12: EXCEPTION HANDLING

- Understanding Error Archetypes (Syntax vs Runtime Errors)
- The Robust Exception Block: try, except, else, and finally
- Raising Exceptions Manually
- Designing Custom User-Defined Exceptions

MODULE 13: FILE INPUT / OUTPUT HANDLING

- File Streams: Reading, Writing, and Appending Data
- Working with Flat Text Files (.txt)
- Working with Structured Spreadsheet Files (.csv)
- Practical File Operations Projects

MODULE 14: OBJECT-ORIENTED PROGRAMMING (OOP)

- Core Concepts: Classes and Objects
- Constructors (__init__) & Instance Variables
- Object Methods & Object Communication
- The Four Pillars of OOP: Encapsulation, Inheritance, Polymorphism, and Abstraction
- Practical OOP Design Projects

Note: *The subsequent advanced and specialized applied-programming modules will be unlocked conditionally, based on the consistency, completion, and evaluation of all assignments by the attendees.*

MODULE 15: ADVANCED PYTHON CONCEPTS

- Decorators (Function Wrappers)
- Generators & Iterators (Memory Optimization)
- Flexible Argument passing: *args and **kwargs
- Deep Dive Comprehensions: List, Dictionary, and Set Comprehension
- Closures & Variable Scopes
- Namespaces & Resolution Rules

MODULE 16: REGULAR EXPRESSIONS (REGEX)

- Introduction to Regular Expression Patterns
- Pattern Matching Techniques
- Advanced Searching & Text Replacement Operations
- Data Validation Examples (Email, Phone, etc.)

MODULE 17: WORKING WITH APIS

- What is an API? Core Concepts
- HTTP Request Methods (GET, POST, PUT, DELETE)
- JSON Format Basics
- Fetching & Parsing Live Data from Web APIs
- API Integration Projects

MODULE 18: DATABASE CONNECTIVITY

- Introduction to Relational Databases
- SQLite Fundamentals
- Performing CRUD Operations (Create, Read, Update, Delete)
- Connecting & Persisting Python Application Data with Databases

MODULE 19: CAPSTONE PYTHON PROJECTS PORTFOLIO

- Interactive Command-Line Calculator
- Number Guessing Game (Logic Training)
- Secure Password Generator
- Student Management System
- Library Management System
- Personal Expense Tracker
- Digital Contact Book
- Interactive Quiz Application

MODULE 20: INTERVIEW & CAREER PREPARATION

- Core Python Interview Questions (Technical Screening)
- Live Coding Challenges & Data Structure Problem Solving
- Advanced Debugging Techniques
- Industry Best Practices & Code Optimization Strategies

MANDATORY CLASS GUIDELINES & RULES

Video Call and Camera Engagement:

Every student must keep their video camera turned on during live sessions. Please ensure your camera is positioned correctly so you are fully visible during the class.

Background and Environment Flexibility:

We understand that you are studying from home. If family members or anyone passes by or moves in the background behind you, it is absolutely fine and there is no problem or issue at all. Do not worry about background movement.

Stable Internet Connection:

Each attendee must ensure they have a strong and stable internet connection before joining the session to prevent any connectivity issues or audio/video lags during live coding.

Active Attendance and Participation:

Continuous attendance is strictly mandatory. If you are called upon or asked any question or explanation during the session, you must respond immediately. Active engagement is required to track your progress.

Mandatory Device for Live Sessions:

All students are strictly required to join the live sessions using a laptop or a desktop system. Joining via mobile devices is highly discouraged as it severely hinders the practical coding experience. If an individual is unable to join through a system on a specific day, a valid and proper reason must be submitted in advance.

Compulsory Practical Setup:

Possessing a functional system for daily practice is an absolute prerequisite for this course. Admission or continued enrollment will not be permitted for candidates without a proper workstation, as lack of practical execution leads to significant learning gaps and disrupts training efficiency.

Minimum System Hardware Requirements:

To ensure seamless performance during environment setup, programming execution, and project deployment, the student's system must meet or exceed the following hardware specifications:

Processor	Intel Core i3 (Minimum 8th Generation) or its equivalent AMD Ryzen processor.
Memory (RAM)	Minimum 8 GB RAM.
Storage	Minimum 256 GB SSD/HDD (Solid State Drive preferred for optimal development speed).