PYTHON PROGRAMMING

Python Programming 🕒 15 hours of content | 40 hours of practice | Exam

1. Getting Started with Python

- What is included in the course
  - Installing Python
  - Writing your first code
  - Downloading and setting up the development environment
  - What is the Python language and how it is used
- Concepts:
  - What are variables and values
  - Working with values
- Data Types:
  - Boolean, integers, floating-point numbers
  - Scientific notation
  - Strings
  - Binary, octal, decimal, and hexadecimal numeral systems
  - Variables
  - Naming conventions
  - Implementing PEP-8 recommendations
- Operators:
  - + - * // % / ** – Numeric operators
  - + – String operators
  - Assignment and shortcut operators
  - Unary and binary operators
  - Priorities and binding
  - << >> | & ^ ~ – Bitwise operators
  - Boolean operators: not, and, or
  - Boolean expressions
  - Relational operators: == != < > <= >=
  - Accuracy of floating-point numbers
  - Type casting
- Additional Concepts:
  - Code clarity
  - Constants and best practices
  - Value comparison using user input
  - Value substitution
  - Conditional statements:
  - if, elif, else, switcher (match-case)
  - Exercises

2. Data Structures

- Working with data structures: list, set, tuple
- Combining and nesting data structures
- Working with dictionaries (Dictionary)
- JSON
- Creating nested structures
- Working with dynamic objects
- Nested/recursive structures
- Exercises

3. Loops and Functions

- Loops (for, while)
  - enumerate
  - continue, break, pass, else
- List Comprehension
- Nested and multiple loops
- Basic functions
  - Built-in functions (sum(), count(), len() etc.)
- Return values
- Using return
  - Using debugger to demonstrate correct function behavior
- Advanced function arguments: *args and **kwargs
- Anonymous functions – lambdas
- Exercises

4. Exception Handling and Multithreading

- Working with exceptions
  - Different exceptions
  - Custom exceptions
- Basic error handling
- Advanced error handling
  - Working with the debugger
  - Debugging process
- try, except, else, finally
- Modules
  - __name__ attribute
  - Importing and splitting files
  - os, sys, random
- Working with the threading module
- Managing threads
- Shared memory and synchronization between threads
- Working with processes (Process)
- Asynchronous programming: async / await
- Exercise

5. Networking and Compilation

- Networking basics
- Working with IP addresses and information
- Basic socket development
- Client/server communication using sockets
- Building basic network tools
- Sharing data over the network
- Communication between multiple users
- Client/server communication including login/authentication
- Secure transmission between users
- TCP vs UDP
- Compiling Python into executable .EXE files
- Exercise

6. Object-Oriented Programming (OOP)

- Introduction to object-oriented programming
- What is a class and what does it represent
- The four OOP principles:
- Inheritance, Encapsulation, Abstraction, Polymorphism
- Creating and working with classes
- Class components: __init__, functions, attributes
- Static and dynamic attributes
- Public and private access
- Inheritance and working with super()
- Multiple inheritance
- Composition vs inheritance
- Building a mini-game using OOP principles

7. Client-Side Development

- Web scraping from websites
- Automated site interaction using Selenium
- Selenium testing and scraping
- Setting up a basic server with Flask
- Fullstack with Flask
- RESTful communication with Flask
- Building a simple website with Flask - Todo list project

8. GUI Development

- Developing graphic user interfaces (GUI)
- Understanding Python’s built-in Tkinter library
- Using Tkinter
- Creating interfaces using HTML and Python
- Developing GUIs with the Eel library
- Developing a “Guess the Word” game
- Developing a network scanning application
- Exercise