Python Complete Online Course

Course Objective

The primary objective of this course is to equip learners with a comprehensive understanding of Python programming, from fundamental concepts to advanced applications. This course is designed to cater to beginners with no prior coding experience, as well as intermediate learners looking to strengthen their programming and software development skills using Python.

By the end of this course, learners will be able to:

- Develop a strong foundation in Core Python, including variables, data types, control structures, functions, file handling, and object-oriented programming.
- Write clean, efficient, and modular Python code following best practices.
- Use Python standard libraries and built-in functions to solve real-world problems.
- Understand and implement advanced Python features such as decorators, generators, error handling, and multithreading.
- Work with external libraries and APIs to perform tasks like web requests, automation, and data serialization.
- Build applications that interact with databases and follow the principles of software design and modular programming.
- Develop simple web applications and REST APIs using popular frameworks like Flask or Django.
- Create unit-tested and well-documented code, suitable for professional and production environments.

The course emphasizes hands-on practice, real-world project development, and a problem-solving approach to ensure learners are industry-ready and confident in using Python across domains such as web development, automation, data processing, and software engineering.

Course Outline

Core Python

1. Introduction to Python

- History of Python
- Features of Python
- Installing Python and IDEs (IDLE, VS Code, PyCharm)
- Writing and running your first Python program
- Python interpreters and virtual environments

2. Python Basics

- Variables, Data Types, and Type Casting
- Input and Output
- Basic Operators (Arithmetic, Relational, Logical)
- Conditional Statements (if, elif, else)
- Looping (for, while)
- range(), enumerate(), zip()
- Comprehensions (list, set, dict)

3. Functions and Modules

- Defining and calling functions
- Arguments (positional, keyword, default, variable-length)
- Return values
- Lambda (anonymous) functions
- Modules and Packages
- import, from, and as

- __name__ == "__main__" in Python

4. Data Structures in Python

- Lists
- Tuples
- Dictionaries
- Sets
- Nested structures
- Common operations and built-in functions

5. String Handling

- String creation and manipulation
- String methods
- String formatting (f-strings, format())
- Regular Expressions (re module)

6. Exception Handling

- Try, Except blocks
- Else and Finally
- Raising exceptions
- Custom exception classes

7. File Handling

- Opening and closing files
- Reading and writing text/binary files
- File methods (read(), write(), seek(), etc.)
- Using with for file handling
- CSV and JSON file handling

8. Object-Oriented Programming (OOP)

- Classes and Objects
- __init__() method (Constructor)
- Instance vs Class variables
- Inheritance (Single, Multiple, Multilevel)
- Method Overriding
- Encapsulation and Polymorphism
- Magic/Dunder methods

9. Python Standard Libraries

- math, datetime, random, os, sys
- collections, itertools, functools
- Environment variables and system operations

10. Working with Virtual Environments

- Creating and managing virtual environments
- Installing and managing packages using pip
- requirements.txt and dependency management

Advanced Python

11. Iterators and Generators

- Custom Iterators
- yield keyword
- Generator expressions

12. Decorators and Closures

- Functions as first-class citizens
- Closures

- Function & Class decorators

13. Advanced OOP Concepts

- Abstract Base Classes (ABC)
- Class methods and static methods
- Property decorators (@property)
- Multiple inheritance and MRO

14. Working with Databases

- Introduction to sqlite3, PostgreSQL/MySQL connectors
- Performing CRUD operations
- Using ORM (SQLAlchemy basics)

15. Python for Web Development (Basics)

- Introduction to Flask or Django
- Routing, templates, and request handling
- Simple REST API with Flask

16. Multithreading and Multiprocessing

- Threading in Python
- GIL (Global Interpreter Lock)
- threading vs multiprocessing
- Process and thread synchronization

17. Working with APIs

- Consuming APIs with requests
- Handling JSON data
- Creating simple REST APIs (Flask/Django REST)

18. Data Serialization

- Working with pickle
- Working with JSON and YAML

19. Unit Testing

- Introduction to unittest and pytest
- Writing and running test cases
- Test discovery and fixtures
- Mocking and test coverage

20. Introduction to Data Science / Automation (Optional)

- pandas, numpy, matplotlib (basics)
- Automating Excel/PDF/Email tasks
- Web scraping with BeautifulSoup or Selenium

Project

This project is designed to reinforce the theoretical and practical knowledge gained throughout the Python Core and Advanced course. While understanding syntax and concepts is essential, true mastery of programming comes through consistent practice and application in solving real-world problems.

The projects that learners will complete here serve as a bridge between classroom learning and industry expectations. They encourage learners to think critically, design efficient solutions, and write maintainable code using Python. Each project has been carefully selected to align with the topics covered in this course.

Total Duration of the Course

- **Total duration:** 2 months.
- **Days:** Monday to Friday (1 hr daily)
- **Mode:** Online

Certificate

- A course completion certificate will be awarded once you have successfully completed this course.

Wishing you a **Smooth Coding** ahead!
