PRAXC INFINITE

AI & TECHNOLOGY EDUCATION

PROFESSIONAL TRAINING PROGRAM

Python Programming Beginner to Intermediate

Build a strong programming foundation and progress to practical, structured Python application development.

LEVEL	DURATION	FORMAT	APPROACH
Beginner to Intermediate	30 Training Hours	Instructor-led training	Practical, hands-on learning

Course Description

Python Programming: Beginner to Intermediate is a structured training program for learners who want to build practical programming skills from the ground up. Participants begin with Python syntax, variables, data types, decision-making, loops, and collections, then progress to functions, modular code, object-oriented programming, file handling, exception handling, and introductory database and API concepts. Guided exercises and a final mini-project help learners apply concepts to practical programming tasks.

Learning Outcomes

• Write Python programs using core syntax, data types, operators, and control flow.

• Work with strings, lists, tuples, sets, and dictionaries to organize information.

• Create reusable functions and structure code into modules.

• Apply foundational object-oriented programming concepts.

• Read and write files and handle common errors safely.

• Use Python packages and virtual environments for basic project setup.

• Understand introductory database and REST API integration concepts.

• Build, test, and present a small Python application.

Who Should Attend

Students, graduates, aspiring Python developers, career starters, and professionals who want a structured introduction to programming. No previous programming experience is required.

Prerequisites & Tools

Prerequisites: Basic computer literacy and a willingness to practise coding.
Tools: Python 3, Visual Studio Code or Jupyter Notebook, and pip/virtual environments. Specific tools may be adjusted to the training environment.

Course Curriculum

The curriculum progresses from core programming concepts to practical application development. Each module includes guided explanation, coding practice, and a learning checkpoint.

Module 1 | Python Fundamentals & Environment Setup

Duration: 2 hours **Module description:** Understand Python, its common applications, and the development workflow. Set up the environment and write your first programs.

Key topics: Python overview; installation and IDE setup; syntax and indentation; comments; variables; input and output.

Module 2 | Data Types, Operators & Strings

Duration: 3 hours **Module description:** Learn how Python represents and processes information, and use operators and string formatting to create useful program logic.

Key topics: Numeric, Boolean, and string types; type conversion; arithmetic, comparison, and logical operators; operator precedence; string methods; f-strings.

Module 3 | Conditions, Loops & Program Flow

Duration: 3 hours **Module description:** Control how a program makes decisions and repeats tasks using clear, structured logic.

Key topics: if/elif/else; nested conditions; for and while loops; range(); break and continue; common loop patterns.

Module 4 | Collections & Data Structures

Duration: 4 hours **Module description:** Store, access, update, and process groups of related values using Python's core collection types.

Key topics: Lists, tuples, sets, and dictionaries; indexing and slicing; collection methods; iteration; nested collections; basic comprehensions.

Module 5 | Functions & Modular Programming

Duration: 4 hours **Module description:** Write reusable code, organise programs into logical units, and reduce duplication through functions and modules.

Key topics: Defining functions; parameters and return values; positional and keyword arguments; default arguments; scope; lambda basics; imports and modules.

Module 6 | Object-Oriented Programming

Duration: 4 hours **Module description:** Introduce the principles used to organise code around objects, making programs easier to maintain and extend.

Key topics: Classes and objects; constructors; attributes and methods; encapsulation; inheritance; method overriding; polymorphism fundamentals.

Module 7 | Files, Exceptions & Project Setup

Duration: 4 hours **Module description:** Build more robust programs by handling errors, working with stored data, and managing a simple Python project.

Key topics: try/except/finally; raising exceptions; text and CSV files; JSON basics; context managers; pip; virtual environments; requirements.txt.

Module 8 | Practical Application & Mini-Project

Duration: 6 hours **Module description:** Bring the course concepts together in a small application, from requirements and implementation through testing and presentation.

Key topics: Project planning; functions and collections; input validation; file-based or SQLite storage; debugging; basic testing; code review and project presentation.

Practical Learning & Assessment

Learning is reinforced through guided coding exercises, module-wise practice tasks, and a mini-project. Suggested projects include a Student Record Manager, Expense Tracker, or Inventory Tracker. Assessment can include practical exercises, code review, and a final project demonstration.

Delivery & Completion

Recommended delivery: Instructor-led sessions with demonstrations, guided practice, and assignments. **Completion:** Course completion may be recognised according to the provider's attendance and assessment policy. A completion certificate, if offered, should be subject to the published course requirements.

COURSE POSITIONING
A practical foundation for learners progressing toward Python development, automation, data analysis, or further study in AI and machine learning.