PRAXC INFINITE

AI & TECHNOLOGY EDUCATION

PROFESSIONAL TRAINING PROGRAM

Professional Python Programming

Beginner to Advanced

12-Week Training Program • 90 Hours
Python Fundamentals, Essential Libraries & Practical Data Analysis

DURATION	LEVEL	TRAINING HOURS	FORMAT
12 Weeks	Beginner to Advanced	90 Hours	Instructor-Led + Practical

Course Description

Build practical programming and data analysis skills through a structured 12-week training program at PraxC Infinite.

Starting with Python fundamentals, the course progressively introduces programming logic, data structures, functions, and object-oriented concepts before moving into numerical computing, data manipulation, and visualization using NumPy, Pandas, Matplotlib, and Seaborn.

Through hands-on exercises, real-world datasets, and a guided project, participants learn to write Python programs, analyse structured data, and communicate findings through meaningful visualizations.

This program provides a practical foundation for learners interested in Python development, data analytics, and further study in Data Science and Machine Learning.

Course Information

Course title: Professional Python Programming | Beginner to Advanced
Duration: 12 Weeks
Total training: 90 Hours
Level: Beginner to Advanced
Prerequisites: No prior programming experience required
Tools: Python 3, Anaconda Navigator, Jupyter Notebook
Libraries: NumPy, Pandas, Matplotlib, Seaborn

Learning Outcomes

• Write Python programs using fundamental programming concepts.

• Solve problems using conditions, loops, functions, and collections.

• Apply basic object-oriented programming principles.

• Read and process files using Python.

• Perform numerical operations using NumPy.

• Clean and analyse datasets using Pandas.

• Create charts and visualizations using Matplotlib.

• Explore statistical patterns using Seaborn.

• Conduct basic exploratory data analysis.

• Develop and present a Python-based data analysis project.

12-Week Detailed Syllabus

The curriculum is organised into 12 progressive modules. The course combines instructor-led explanation, coding demonstrations, practical exercises, and a final project assessment.

Module 1: Introduction to Python Programming

Module description: Understand Python programming fundamentals and configure the development environment. Learn basic syntax, variables, data types, and input/output operations while writing your first programs.

Topics covered:

• Introduction to Python and its applications

• Installing Python and configuring the environment

• Anaconda Navigator and Jupyter Notebook

• Python syntax, indentation, and comments

• Variables, naming conventions, and data types

• Input, output, and basic calculations

Module 2: Operators & Conditional Statements

Module description: Learn how to process information and control program execution using operators and conditional statements. Develop logical thinking through programs that evaluate conditions and make decisions.

Topics covered:

• Arithmetic, comparison, and logical operators

• Assignment, membership, and identity operators

• Type conversion and operator precedence

• String operations and formatting

• if, elif, and else statements

• Nested conditions and compound expressions

Module 3: Loops & Problem-Solving

Module description: Develop programming logic using loops and iteration. Learn to automate repetitive tasks and solve common problems using structured control flow.

Topics covered:

• for and while loops

• range() and loop control

• Nested loops

• break, continue, and pass

• Loop-based calculations and patterns

• Introduction to list comprehensions

Module 4: Python Data Structures

Module description: Learn to organise, access, and manipulate collections of information using Python's built-in data structures. Understand how to select appropriate structures for different programming requirements.

Topics covered:

• Lists and list operations

• Tuples and immutable data

• Sets and unique elements

• Dictionaries and key-value pairs

• Indexing, slicing, and collection methods

• Nested collections, sorting, and iteration

Module 5: Functions & Modular Programming

Module description: Develop reusable and organised Python programs using functions and modules. Learn to divide tasks into smaller components and improve code readability through structured programming.

Topics covered:

• Defining functions and return values

• Parameters and arguments

• Default arguments and *args / **kwargs

• Variable scope

• Lambda functions and recursion fundamentals

• Built-in functions and importing modules

Module 6: Object-Oriented Python & File Handling

Module description: Explore object-oriented programming fundamentals and learn to store and retrieve information using files. Apply these concepts to build structured programs that manage information effectively.

Topics covered:

• Classes and objects

• Constructors, attributes, and methods

• Basic inheritance and encapsulation

• Reading and writing text files

• CSV and JSON fundamentals

• Basic exception handling

Module 7: Numerical Computing with NumPy

Module description: Discover how NumPy supports numerical computing and array-based operations. Learn to create and manipulate arrays and perform common mathematical and statistical calculations.

Topics covered:

• Introduction to NumPy and arrays

• Array creation, dimensions, and shapes

• Indexing and slicing

• Reshaping arrays

• Vectorised mathematical operations

• Aggregation and basic statistical functions

Module 8: Data Analysis with Pandas

Module description: Learn to work with structured datasets using Pandas. Practise importing, inspecting, cleaning, transforming, and summarising tabular data.

Topics covered:

• Introduction to Series and DataFrames

• Importing CSV and Excel datasets

• Inspecting and summarising data

• Selecting, filtering, and sorting records

• Handling missing and duplicate values

• Grouping, aggregation, and exporting data

Module 9: Data Visualization with Matplotlib

Module description: Learn to communicate information visually using Matplotlib. Create clear charts to explore trends, compare categories, and understand numerical distributions.

Topics covered:

• Introduction to Matplotlib

• Line charts and bar charts

• Scatter plots and histograms

• Figure and axis fundamentals

• Titles, labels, legends, and chart formatting

• Saving visualizations

Module 10: Statistical Visualization with Seaborn

Module description: Explore Seaborn's statistical visualization tools to identify patterns, distributions, and relationships in datasets.

Topics covered:

• Introduction to Seaborn

• Distribution plots

• Count plots and categorical charts

• Box plots and scatter plots

• Heatmaps and correlation visualization

• Chart styling and interpretation

Module 11: Practical Data Analysis & Exploratory Data Analysis

Module description: Combine Python and its data libraries to explore datasets, assess data quality, identify trends, and prepare meaningful visual summaries.

Topics covered:

• Introduction to exploratory data analysis

• Dataset inspection and quality checks

• Data cleaning and preprocessing

• Handling missing and duplicate records

• Descriptive statistics and grouping

• Identifying trends and patterns

• Combining NumPy and Pandas

• Visualising findings with Matplotlib and Seaborn

Module 12: Project & Assessment

Module description: Apply the knowledge gained throughout the course to complete a practical Python-based data analysis project. Participants will clean a dataset, perform analysis, generate visualizations, and present their findings.

Topics covered:

• Project planning and dataset selection

• Data loading and inspection

• Data cleaning and transformation

• Numerical analysis

• Data visualization

• Interpreting results

Expected outcome: Complete and present a Python-based data analysis project demonstrating data cleaning, analysis, visualization, and interpretation.

Training Methodology & Assessment

Training methodology: instructor-led explanations, live coding demonstrations, guided exercises, and practical assignments.

Assessment may include coding exercises, module-based tasks, and the final project presentation. Any certificate or completion criteria should follow the training provider's published policy.

PROGRAM SUMMARY
Professional Python Programming | Beginner to Advanced
12 Weeks • 90 Training Hours • Python Fundamentals • NumPy • Pandas • Matplotlib • Seaborn • Project & Assessment