

Beginning Python — Five-Day Foundations

Core programming, data structures, functions, objects and files

FA-0784 · 5 days (core programme)

Software Development

Introduction

Build Python programming foundations without needing prior programming experience. Translate simple problems into logical steps, algorithms and pseudocode, then implement functions, modules, objects and file-based programs. Experienced programmers or returning Python users can also use the course to refresh the fundamentals. Optional extensions are separate from the five-day core.

Learning outcomes

- Use a selected Python editor, notebook or command-line environment.
- Apply syntax, operators, conditions, loops and core data structures.
- Develop reusable functions and modules with appropriate arguments and return values.
- Create classes and apply inheritance, properties and introductory polymorphism.
- Read/write files and work with directories.
- Organise and document small programs and identify resources for further learning.

Prerequisites

- Ability to use a computer
- Ability to use the internet
- Manage files and understand basic file systems in the computer
- Ability to communicate in the English language

A suitable computer, browser, reliable internet and a compatible supported Python environment.

Training outline

1. Day 1 — Python and development tools

- Python history, features and application areas.
- Install a compatible Python version on Windows, macOS or Linux.
- Compare Jupyter Notebook, VS Code and PyCharm; use one selected environment for practice.
- Create/manage notebooks or run/debug scripts; use the CLI and Python REPL.

2. Day 1 — Syntax and control flow

- Indentation, variables, data types and conversion.
- Arithmetic, comparison and logical operators.
- if/elif/else, for/while, break, continue and pass.

3. Day 2 — Core data structures

- Lists: creation, access, methods and comprehensions.
- Tuples: creation, access and operations.
- Sets: creation, membership and operations.
- Dictionaries: keys/values, access, methods and functions.

4. Day 3 — Functions and modules

- Function definition, parameters, arguments and return values.
- Import modules and use their functions and variables.
- Explore standard-library modules and build a small standalone module.

5. Day 4 — Object-oriented programming

- OOP concepts, classes, objects, instance variables and methods.
- Inheritance, overriding and introductory multiple inheritance.
- Encapsulation conventions and name mangling; Python does not enforce private/protected access like some other languages.
- Properties with getter, setter and deleter behaviour.

- Polymorphism, operator overloading and abstract base classes.
- Flexible method arguments and overriding, rather than automatic overload selection from multiple same-name definitions.

6. Day 5 — File handling and good practice

- Open/close, read/write and file modes.
- Create/delete/list directories and manage the working directory in a prepared lab.
- Naming conventions, code organisation, comments and documentation.
- Find official documentation, tutorials and community resources for further learning.

Optional extension topics

Select appropriate extensions according to learning goals and available additional time; the five-day core does not promise all of the following.

- Errors and exceptions: raise, try, except and finally.
- Regular expressions: pattern matching, search/replace and flags.
- Networking: sockets, HTTP requests, permitted scraping and REST APIs.
- Concurrency: threading and multiprocessing concepts.
- Library/framework overview: NumPy, SciPy, pandas, Matplotlib, Seaborn, TensorFlow, PyTorch, Flask and Django.
- Virtual environments: create/activate and install packages.
- Packaging/distribution and application-deployment concepts.