

**Advanced Python Programming**

**Course Number:** PYTH-102  
**Duration:** 4 days

**Overview**

Accelebrate's Advanced Python Programming training course picks up where Introduction to Python Programming left off, covering some topics in more detail and adding new ones. For instance, classes are covered in greater detail, with new coverage of OS services, date/time management, binary data, unit testing, database connectivity, network programming, and more.

**Prerequisites**

All students should be able to write simple Python scripts using basic data types, program structures, and the standard Python library.

**Materials**

In addition to the course manual, students will receive a Python quick reference.

**Software Needed on Each Student PC**

* Any Windows, Linux, or macOS operating system
* Python 3.x installed (Anaconda bundle recommended)
* An IDE with Python support (PyCharm Community Edition is an excellent free option, but there are several other good ones)

**Objectives**

* Leverage OS services
* Add enhancements to classes
* Code graphical interfaces for applications
* Understand advanced Python metaprogramming concepts
* Create easy-to-use and easy-to-maintain modules and packages
* Implement  and run unit tests
* Create multithreaded and multi-process applications
* Interact with network services
* Design professional scripts
* Query databases

**Outline**

* Introduction
* Python Refresher
  + Built-in data types
  + Lists and tuples
  + Dictionaries and sets
  + Program structure
  + Files and console I/O
  + If statement
  + for
  + Built-in functions
  + User-defined functions
  + Modules and packages
  + Basic OOP
* OS Services
  + The os and os.path modules
  + Environment variables
  + Launching external commands with subprocess
  + Walking directory trees
  + Paths, directories, and filenames
  + Working with file systems
* Dates and Times
  + Basic date and time classes
  + Different time formats
  + Converting between formats
  + Formatting dates and times
  + Parsing date/time information
* Binary Data
  + What is Binary Data?
  + Binary vs text
  + Using the Struct module
* Pythonic Programming
  + The Zen of Python
  + Tuples
  + Advanced unpacking
  + Sorting
  + Lambda functions
  + List comprehensions
  + Generator expressions
  + String formatting
* Functions, Modules, and Packages
  + Four types of function parameters
  + Four levels of name scoping
  + Single/multi-dispatch
  + Relative imports
  + Using \_\_init\_\_ effectively
  + Documentation best practices
* Enhancing Classes
  + Class/static data and methods
  + Inheritance (or composition)
  + Abstract base classes
  + Creating attributes with attr
  + Implementing protocols (context, iterator, etc.)
* Metaprogramming
  + Implicit properties
  + globals() and locals()
  + Working with object attributes
  + The inspect module
  + Callable classes
  + Decorators
  + Monkey patching
* Developer Tools
  + Analyzing programs with pylint
  + Using the debugger
  + Profiling code
  + Testing speed with benchmarking
* Unit Testing with PyTest
  + What is a unit test
  + Creating test cases
  + Writing and running tests
  + Test harnesses
  + Working with fixtures
* Database Access
  + The DB API
  + Available Interfaces
  + Connecting to a server
  + Creating and executing a cursor
  + Fetching data
  + Parameterized statements
  + Using Metadata
  + Transaction control
  + ORMs and NoSQL overview
* PyQt
  + Overview
  + Qt Architecture
  + Using designer
  + Standard widgets
  + Event handling
  + Extras
* Network Programming
  + Built-in classes
  + Using requests
  + Grabbing web pages
  + Sending email
  + Working with binary data
  + Consuming RESTful services
  + Remote access (SSH)
* Multiprogramming
  + The threading module
  + Sharing variables
  + The queue module
  + The multiprocessing module
  + Creating pools
  + About async programming
* Scripting for System Administration
  + Running external programs
  + Parsing arguments
  + Creating filters to read text files
  + Logging
* Serializing Data
  + Working with XML
  + XML modules in Python
  + Getting started with ElementTree
  + Parsing XML
  + Updating an XML tree
  + Creating a new document
  + About JSON
  + Reading JSON
  + Writing JSON
  + Reading/writing CSV files
  + YAML, other formats as time permits
* Advanced Data Handling [as time permits]
  + Discover the collections module
  + Use defaultdict, Counter, and namedtuple
  + Create dataclasses
  + Store data offline with pickle
* Type Hinting [as time permits]
  + Annotate variables
  + Learn what type hinting does NOT do
  + Use the typing module for detailed type hints
  + Understand
  + Write stub interfaces
* Conclusion