skillsoft global knowledge™

ADVANCED PYTHON

Course Code: 821512

Understand Python’s capabilities beyond basic syntax in this hands-on advanced-level course.

This course will help you gain an understanding of Python's capabilities beyond basic syntax with a focus on widely accepted Pythonic constructs and procedures that will enable you to write reliable, optimized, and modular applications. This very hands-on course includes a deep dive into Pythonic data structures, exception handling, meta programming, regular expression, advanced file-handling, asynchronous programming, and more. At the completion of the course, you will also gain an understanding of unit testing in Python with lab-based practices designed to help you create and run unit test cases.

What You'll Learn

This course has 50% hands-on labs to 50% lecture ratio with engaging instruction, demos, group discussions, labs, and project work in which you’ll learn:

- Enhancements to classes
- Advanced Python metaprogramming concepts
- Writing robust code using exception handling
- Working with different data structures supported in Python
- Search and replace text with regular expressions
- Easy-to-use and easy-to-maintain modules and packages
- Creating multithreaded and multi-process applications
- Implementing and execute unit tests

Who Needs to Attend

This course is designed for students with Python programming literacy who want to learn about advanced Python features and how to automate and simplify tasks.

Prerequisites

Students should have experience writing Python scripts, as well as a user-level knowledge of Unix/Linux, Mac, or Windows.

skillsoft global knowledge™

ADVANCED PYTHON

Course Code: 821512

CLASSROOM LIVE	$2,495 USD	5 Day

Classroom Live Outline

Day 1

1. **Python refresher**
   - Built-in data types
   - Lists and tuples
   - Dictionaries and sets
   - Program structure
   - Files and console I/O
   - If statement
   - for and while loops
2. **Data Structures and Algorithms**
   - Linked list
   - Stack
   - Queue
   - Trees
   - Graphs
   - Sorting algorithms

Day 2

1. **Errors and Exception Handling**
   - Syntax errors
   - Exceptions
   - Using try/catch/else/finally
   - Handling multiple exceptions
   - Ignoring exceptions
2. **Implementing Regular Expressions**
   - RE Objects
   - Searching and matching
   - Using Regular Expression to search data sets

- Searching for data in Wireshark Traces (Python and *.pcaps)
- Compilation flags
- Groups and special groups
- Replacing text
- Splitting strings

3. **Advanced Functional Features of Python**
    - Advanced unpacking
    - List Comprehension
    - Anonymous functions
    - Lambda expressions
    - Generator Expression
    - Decorator
    - Closure
    - Single/multi dispatch
    - Relative imports
    - Using __init__ effectively
    - Documentation best practices

Day 3

1. **Metaprogramming**
    1. OOP conventions
    2. Class/static data and methods
    3. Parse information to create classes using a dictionary
    4. Super() method
    5. Metaclasses
    6. Abstract base classes
    7. Implementing protocols (context, iterator, etc.) with special methods
    8. Implicit properties
    9. Globals() and locals()
    10. Working with object attributes
    11. The inspect module
    12. Callable classes
    13. Monkey patching
2. **Advanced file handling**
    - Paths, directories, and filenames
    - Checking for existence
    - Permissions and other file attributes
    - Walking directory trees
    - Creating filters with fileinput
    - Using shutil for file operations

Day 4

1. **Advanced Data Structure features in Python**
    - Use defaultdict, Counter, and namedtuple
    - Create data classes
    - Store data offline with pickle

- Pretty printing data structures
- Compressed archives (zip, gzip, tar, etc.)
- Persistent data

2. **Multiprogramming**
    - Concurrent programming
    - Multithreading
    - The threading module
    - Sharing variables
    - The queue module
    - The multiprocessing module
    - Creating pools
    - Coroutines
    - About async programming
3. **Python Design Patterns**
    - Need for design patterns and types
    - Creational
    - Structural
    - Behavioral
    - Best coding practices

Day 5

1. **Developer Tools**
    - Analyzing programs with pylint
    - Using the debugger
    - Profiling code
    - Testing speed with benchmarking
2. **Unit testing with PyTest**
    - What is a unit test
    - Testing with Unit-test framework
    - Testing with PyTest
    - Testing with doctest
    - Writing tests
    - Working with fixtures
    - Test runners
    - Mocking resources
3. **Writing real-life applications**
    - Build the classic minesweeper game in the command line
    - Build a program that can go into any folder on your computer and rename all of the files based on the conditions set in your Python code
    - Implement the binary search algorithm
    - Build a random password generator
    - Build a countdown timer using the time Python module.

Classroom Live Labs

About 50% of the content of this very hands-on course is lab-based practice.

skillsoft global knowledge™

ADVANCED PYTHON

Course Code: 821512

VIRTUAL CLASSROOM LIVE	$2,495 USD	5 Day

Virtual Classroom Live Outline

Day 1

1. **Python refresher**
   - Built-in data types
   - Lists and tuples
   - Dictionaries and sets
   - Program structure
   - Files and console I/O
   - If statement
   - for and while loops
2. **Data Structures and Algorithms**
   - Linked list
   - Stack
   - Queue
   - Trees
   - Graphs
   - Sorting algorithms

Day 2

1. **Errors and Exception Handling**
   - Syntax errors
   - Exceptions
   - Using try/catch/else/finally
   - Handling multiple exceptions
   - Ignoring exceptions
2. **Implementing Regular Expressions**
   - RE Objects
   - Searching and matching
   - Using Regular Expression to search data sets

- Searching for data in Wireshark Traces (Python and *.pcaps)
- Compilation flags
- Groups and special groups
- Replacing text
- Splitting strings

3. **Advanced Functional Features of Python**
    - Advanced unpacking
    - List Comprehension
    - Anonymous functions
    - Lambda expressions
    - Generator Expression
    - Decorator
    - Closure
    - Single/multi dispatch
    - Relative imports
    - Using __init__ effectively
    - Documentation best practices

Day 3

1. **Metaprogramming**
    1. OOP conventions
    2. Class/static data and methods
    3. Parse information to create classes using a dictionary
    4. Super() method
    5. Metaclasses
    6. Abstract base classes
    7. Implementing protocols (context, iterator, etc.) with special methods
    8. Implicit properties
    9. Globals() and locals()
    10. Working with object attributes
    11. The inspect module
    12. Callable classes
    13. Monkey patching
2. **Advanced file handling**
    - Paths, directories, and filenames
    - Checking for existence
    - Permissions and other file attributes
    - Walking directory trees
    - Creating filters with fileinput
    - Using shutil for file operations

Day 4

1. **Advanced Data Structure features in Python**
    - Use defaultdict, Counter, and namedtuple
    - Create data classes
    - Store data offline with pickle

- Pretty printing data structures
- Compressed archives (zip, gzip, tar, etc.)
- Persistent data

2. **Multiprogramming**
    - Concurrent programming
    - Multithreading
    - The threading module
    - Sharing variables
    - The queue module
    - The multiprocessing module
    - Creating pools
    - Coroutines
    - About async programming
3. **Python Design Patterns**
    - Need for design patterns and types
    - Creational
    - Structural
    - Behavioral
    - Best coding practices

Day 5

1. **Developer Tools**
    - Analyzing programs with pylint
    - Using the debugger
    - Profiling code
    - Testing speed with benchmarking
2. **Unit testing with PyTest**
    - What is a unit test
    - Testing with Unit-test framework
    - Testing with PyTest
    - Testing with doctest
    - Writing tests
    - Working with fixtures
    - Test runners
    - Mocking resources
3. **Writing real-life applications**
    - Build the classic minesweeper game in the command line
    - Build a program that can go into any folder on your computer and rename all of the files based on the conditions set in your Python code
    - Implement the binary search algorithm
    - Build a random password generator
    - Build a countdown timer using the time Python module.

Virtual Classroom Live Labs

About 50% of the content of this very hands-on course is lab-based practice.

Jun 9 - 13, 2025 | 8:30 AM - 4:30 PM EDT

ADVANCED PYTHON

Course Code: 821512

PRIVATE GROUP TRAINING	5 Day

Visit us at www.globalknowledge.com or call us at 1-866-716-6688.

Date created: 5/9/2025 12:55:31 AM