Advanced Python

CompuEducación®

55285 Advanced Python

Course length: 2 Days

About this course:

In this Python training course, students already familiar with Python programming will learn advanced Python techniques.

This advanced Python course is taught using Python 3; however, differences between Python 2 and Python 3 are noted.

At course completion:

At course completion

Work with the Collections module.

Understand mapping and filtering and lambda functions.

Perform advanced sorting.

Work with regular expressions in Python.

Work with databases, CSV files, JSON, and XML.

Write object-oriented code in Python.

Test and debug your Python code.

Understand Unicode and text encoding.

Requisites:

Experience in the following is required for this Python class:

Basic Python programming experience. In particular, you should be very comfortable with:

Working with strings.

Working with lists, tuples and dictionaries.

Loops and conditionals.

Writing your own functions.

Experience in the following would be useful for this Python class:

Some exposure to HTML, XML, JSON, and SQL.

TEMARIO

Module 1: Advanced Python Concepts

In this lesson, you will learn about some Python functionality and techniques that are commonly used but require a solid foundation in Python to understand.

Lessons:
Lambda Functions
Advanced List Comprehensions
Collections Module
Mapping and Filtering
Mutable and Immmutable Built-in Objects
Sorting
Unpacking Sequences in Function Calls

Lab 1: Exercises in this Lesson
Rolling Five Dice

- Creating a defaultdict
- Creating a OrderedDict
- Creating a Counter
- Working with a deque
- Converting list.sort() to sorted(iterable)
- Converting a String to a datetime.date Object

After completing this module, students will be able to:

- Work with lambda functions.
- Write more advanced list comprehensions.
-
- Work with the collections module to create named tuples, defaultdicts, ordereddicts, counters, and deques.
-
- Use mapping and filtering.
- Sort sequences.
- Unpack sequences in function calls.
- Create modules and packages.

Module 2: Working with Data

Data is stored in many different places and in many different ways. There are Python modules for all of the most common ways.
Lessons
Relational Databases CSV
Getting Data from the Web JSON

Lab 1: Exercises in this Lesson

- Querying a SQLite Database
- Inserting File Data into a Database
- Comparing Data in a CSV File
- Requests and Beautiful Soup
- Using JSON to Print Course Data
- After completing this module, students will be able to:
- Access and work with data stored in a relational database.
- Access and work with data stored in a CSV file.
- Get data from a web page.
- Access and work with data stored as HTML and XML.
- Access an API
- Access and work with data stored as JSON.

Module 3: Testing and Debugging

This module explains how to test and debug using Python

Lessons

Testing for Performance
The unittest Module

Lab 1: Exercises in this Lesson

Fixing Functions

After completing this module, students will be able to:

Test performance with timers and using the timeit module.

To write unit tests using the unittest module.

Module 4: Classes and Objects

An object is something that has attributes and/or behaviors, meaning it is certain ways and does certain things. In the real world, everything could be considered an object. Some objects are tangible, like rocks, trees, tennis racquets, and tennis players. And some objects are intangible, like words, colors, tennis swings, and tennis matches.

Lessons

- Attributes
- Behaviors
- Classes vs. Objects
- Attributes and Methods
- Private Attributes
- Properties
- Documenting Classes
- Inheritance
- Static Methods
- Class Attributes and Methods
- Abstract Classes and Methods
- Understanding Decorators

Lab 1: Exercises in this Lesson

- Adding a roll() Method to Die
- Properties
- Documenting the Die Class
- Extending to Die Class
- Extending the roll() Method
- After completing this module, students will be able to:
- Create classes and objects in Python.
- Write instance methods, class methods, and static methods.
- Define properties.
- Create subclasses using inheritance.
- Create abstract classes.
- Appropriately document Python classes.
- Understand how decorators work.