Mastering Python Programming (with Next-Level Topics) (TTPS4820)

Modality: Virtual Classroom

Duration: 5 Days

About this course:

Learning Python Programming is an easy, practical, hands-on Python training program designed for intermediate users, which takes students from the fundamentals of running and writing Python scripts to more powerful features like regular expressions, file operations, working with binary data, and utilizing the comprehensive features of Python modules. Extra focus is placed on Python-specific features, like output formatting, array slices, and tuples.

A practical, comprehensive course offers an in-depth analysis of working with the language of programming, not syntax and grammar academic summary. Participants should be able to use Python quickly to fulfill real-world tasks.

Course Objective:

Participants will be led during the course through a sequence of increasingly specialized topics, where each subject consists of group discussion, lecture, detailed hands-on laboratory experiments, and lab analysis. This training is "expertise-centric," designed to train participants in critical Python and web development aptitudes, combined with best practices from the most recent, successful techniques.

Participants will discover:

- Explaining Functions
- Getting Started
- Use of Modules
- Overview of Python Classes
- Runs scripts in Python
- Sets and Dictionaries
- Exception and Errors Handling
- Data Sequence
- Control of Flow
- Advanced data types
- Work with Files
- Times and dates
- Standard Library Highlights
- Network services
- Real-life programming
- Regular Expressions

Work in a dedicated, hands-on studying environment, directed by our Python professional specialist, participants can learn to:

- Allow effective use of python data types
- Inform yourself about the standard library and its work-saving modules
- Work with calendars, times, and dates
- Build Python scripts running according to best practices
- Comprehend Pythonic functions, including understandings and iterators
- Read and write text and binary data files
- Build professional, "real-world" Python apps
- Compose robust code for error handling
- Check and replace normal expressions with the text
- Using lesser-known but potent forms of Python data
- Learn when to make use of collections like dictionaries, lists, and sets

Audience:

This training is ideal for experienced users, website administrators, and system administrators who are using Python to assist their server installations, as well as anyone else who needs to use Python scripts to simplify or automate specific tasks.

Prerequisite:

Participants will also have an understanding of Linux/Unix, Windows, or Mac at work, user-level. Although not needed, fundamental skills would be useful in at least one certain programming language.

Course Outline:

Module 1: An Overview of Python

- What is python?
- An overview of Python
- What is python?
- Python Timeline
- Advantages/Disadvantages of Python
- Getting help with pydoc

Module 2: The Python Environment

- Starting Python
- Using the interpreter
- Running a Python script
- Python scripts on Unix/Windows
- Editors and IDEs

Module 3: Getting Started

- Using variables
- Builtin functions
- Strings

- Numbers
- Converting among types
- Writing to the screen
- Command line parameters

Module 4: Flow Control

- About flow control
- White space
- Conditional expressions
- Relational and Boolean operators
- While loops
- Alternate loop exits

Module 5: Sequences

- About sequences
- Lists and list methods
- Tuples
- Indexing and slicing
- Iterating through a sequence
- Sequence functions, keywords, and operators
- List Comprehensions
- Generator Expressions
- Nested sequences

Module 6: Working with files

- File Overview
- Opening a text file
- Reading a text file
- Writing to a text file
- Reading and writing raw (binary) data
- Converting binary data with struct

Module 7: Dictionaries and Sets

- About dictionaries
- Creating dictionaries
- Iterating through a dictionary
- About sets
- Creating sets
- Working with sets

Module 8: Functions

- Defining functions
- Parameters

- Global and local scope
- Nested functions
- Returning values

Module 9: Sorting

- The sorted() function
- Alternate keys
- Lambda functions
- Sorting collections
- Using operator.itemgetter()
- Reverse sorting

Module 10: Errors and Exception Handling

- Syntax errors
- Exceptions
- Using try/catch/else/finally
- Handling multiple exceptions
- Ignoring exceptions

Module 11: Modules and Packages

- The import statement
- Module search path
- Creating Modules
- Using packages
- Function and Module aliases

Module 12: Classes

- About o-o programming
- Defining classes
- Constructors
- Methods
- Instance data
- Properties
- Class methods and data

Module 13: Regular Expressions

- RE syntax overview
- RE Objects
- Searching and matching
- Compilation flags
- Groups and special groups
- Replacing text
- Splitting strings

Module 14: The standard library

- The sys module
- Launching external programs
- Math functions
- Random numbers
- The string module
- Reading CSV data

Module 15: Dates and times

- Working with dates and times
- Translating timestamps
- Parsing dates from text
- Formatting dates
- Calendar data

Module 16: Working with the file system

- Paths, directories, and filenames
- Checking for existence
- Permissions and other file attributes
- Walking directory trees
- Creating filters with file input
- Using shutil for file operations

Module 17: Advanced data handling

- Defaultdict and Counter
- Prettyprinting data structures
- Compressed archives (zip, gzip, tar, etc.)
- Persistent data

Module 18: Network services

- Grabbing web content
- Sending email
- Using SSH for remote access
- Using FTP

Module 19: Writing real-life applications

- Parsing command-line options
- Detecting the current platform
- Trapping signals
- Implementing logging
- Python Timeline
- Advantages/Disadvantages of Python

- Getting help with pydoc