Document Generated: 10/26/2025

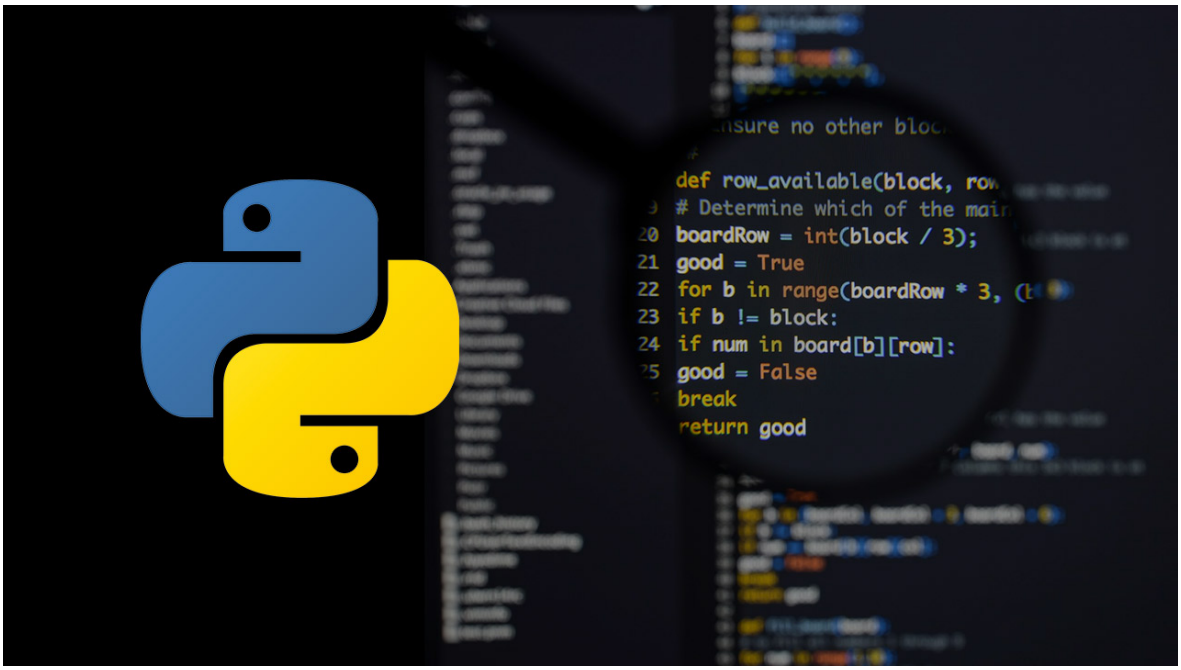
Learning Style: Virtual Classroom

Technology:

Difficulty: Intermediate

Course Duration: 5 Days

Applied Python for Scientists and Engineers (TTPS4870)



About this course:

Geared for scientists and engineers with potentially light practical programming background or experience, ***Applied Python for Scientists and Engineers*** is a hands-on Python course that provides a ramp-up to using Python for scientific and mathematical computing. Students will explore basic Python scripting skills and concepts, and then move to the most important Python modules for working with data, from arrays, to statistics, to plotting results.

The average salary of a Python Data Scientist is **$93,185** per year.

Course objective:

Working within in an engaging, hands-on learning environment, guided by our expert Python practitioner, attendees will learn to use Python to:

- Create and run basic programs
- Design and code modules and classes
- Implement and run unit tests
- Use benchmarks and profiling to speed up programs
- Process XML and JSON
- Manipulate arrays with **numpy**
- Get a grasp of the diversity of subpackages that make up **scipy**
- Use iPython notebooks for ad hoc calculations, plots, and what-if?
- Manipulate images with PIL
- Solve equations with sympy
- Get introduced to advanced data science with Python including SciKit-Learn and PySpark

Audience:

- While there are no specific programming prerequisites, basic programming experience would be helpful.

Prerequisite:

- Students should be comfortable working with files and folders, and should not be afraid of the command line.

Course Outline:

Module 1: The Python environment

- About Python
- Starting Python
- Using the interpreter
- Running a Python script
- Python scripts on UNIX/Windows
- Using the Spyder editor

Module 2: Getting started

- Using variables
- Built-in functions
- Strings
- Numbers

- Converting among types
- Writing to the screen
- String formatting
- Command line parameters

Module 3: Flow control

- About flow control
- White space
- Conditional expressions (if,else)
- Relational and Boolean operators
- While loops
- Alternate loop exits

Module 4: Sequences

- About sequences
- Lists and tuples
- Indexing and slicing
- Iterating through a sequence
- Sequence functions, keywords, and operators
- List comprehensions
- Generator expressions
- Nested sequences

Module 5: Working with files

- File overview
- Opening a text file
- Reading a text file
- Writing to a text file
- Raw (binary) data

Module 6: Dictionaries and sets

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

Module 7: Functions

- Defining functions
- Parameters
- Variable scope
- Returning values
- Lambda functions

Module 8: Errors and exception handling

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

Module 9: OS services

- The O module
- Environment variables
- Launching external commands
- Walking directory trees
- Paths, directories, and filenames
- Working with file systems
- Dates and times

Module 10: Pythonic idioms

- Small Pythonisms
- Lambda functions
- Packing and unpacking sequences
- List Comprehensions
- Generator Expressions

Module 11: Modules and packages

- Initialization code
- Namespaces
- Executing modules as scripts
- Documentation
- Packages and name resolution
- Naming conventions
- Using imports

Module 12: Classes

- Defining classes
- Constructors
- Instance methods and data
- Attributes
- Inheritance
- Multiple inheritance

Module 13: XML and JSON

- Using ElementTree
- Creating a new XML document
- Parsing XML
- Finding by tags and XPath
- Parsing JSON into Python

- Parsing Python into JSON

Module 14: iPython and Jupyter

- iPython basics
- Terminal and GUI shells
- Creating and using notebooks
- Saving and loading notebooks
- Ad hoc data visualization

Module 15: numpy

- numpy basics
- Creating arrays
- Indexing and slicing
- Large number sets
- Transforming data
- Advanced tricks

Module 16: scipy

- What can scipy do?
- Most useful functions
- Curve fitting
- Modeling
- Data visualization
- Statistics

Module 17: A tour of scipy subpackages

- Clustering
- Physical and mathematical Constants
- FFTs
- Integral and differential solvers
- Interpolation and smoothing
- Input and Output
- Linear Algebra
- Image Processing
- Distance Regression
- Root-finding
- Signal Processing
- Sparse Matrices
- Spatial data and algorithms
- Statistical distributions and functions

Module 18: pandas

- pandas overview
- Dataframes
- Reading and writing data

- Data alignment and reshaping
- Fancy indexing and slicing
- Merging and joining data sets

Module 19: Seaborn

- Creating a basic plot
- Commonly used plots
- Ad hoc data visualization
- Exporting images

Module 20: matplotlib

- Creating a basic plot
- Commonly used plots
- Ad hoc data visualization
- Advanced usage
- Exporting images

Module 21: The Python Imaging Library (PIL) [Optional]

- PIL overview
- Core image library
- Image processing
- Displaying images

Module 22: SciKit-Learn Essentials [Optional]

- SciKit overview
- SciKit-Learn overview
- Algorithms Overview
- Classification, Regression, Clustering, and Dimensionality Reduction
- SciKit Demo

Module 23: PySpark Overview [Optional]

- Python and Spark
- SciKit-Learn vs. Spark MLlib
- Python at Scale
- PySpark Demo

Credly Badge:

Display your Completion Badge And Get The Recognition You Deserve.

Add a completion and readiness badge to your Linkedin profile, Facebook page, or Twitter account to

validate your professional and technical expertise. With badges issued and validated by Credly, you can:

- Let anyone verify your completion and achievement by clicking on the badge
- Display your hard work and validate your expertise
- Display each badge's details about specific skills you developed.

Badges are issued by QuickStart and verified through Credly.

Find Out More or See List Of Badges