# Python Programming for Security Professionals (TTPS4890)

**Modality: Virtual Classroom**

**Duration: 5 Days**

## About this course:

Intended for experienced security experts, this class is a practical, introductory, hands-on Python instructional class that leads the understudy from the nuts and bolts of running and writing Python contents to further developed features, for example, working with binary data, regular expressions, file operations, and utilizing the broad usefulness of Python modules. Additional emphasis is put on features one of a kind to Python, for example, array slices, tuples, and output formatting. This far-reaching, pragmatic course gives a top to bottom exploration of functioning with the programming language, not grammar and syntax academic overview.

This course is customized explicitly for the Analysts of Security and other people who need to utilize Python usefulness for security-related assignments, for example, log forensics or manipulation. This course, which tends to the practices of secure coding, is fundamental for security experts that are performing security audits and reviews of Python applications or are supporting advancement groups in executing better resistances in Python.

The normal pay of a Python developer is $116,379 every year.

## Course Objective:

All through the course, understudies will be driven through a progression of dynamically propelled topics, where every point comprises of group discussion, lecture, lab review, and comprehensive hands-on lab exercises. This course is intended to prepare participants in web development and core Python abilities beyond an intermediate level, with the most up-to-date, effective strategies with best practices.

Working inside a hands-on learning and engaging environment that directed by our master Python expert, learners will figure out how to:

- Write and Read files with both binary data and text.
- Properly Use python data types.
- Find and replace the content with regular expressions.
- Make "real-world", proficient applications of Python
- Make operational Python contents following best procedures
- Utilize incredible Python data types
- Get acquainted with the standard library and its work-sparing modules
- Compose robust code by means of handling exception.
- Realize when to utilize collections such as dictionaries, lists, and sets.
- Work with times, dates, and calendars.
- Understand the features of Python such as iterators and comprehensions.

## Audience:

This course is suitable for system administrators, advanced users, and web site administrators who need to utilize Python to help their installations of servers, also any other person who needs to simplify or automate common assignments with the utilization of Python contents.

## Prerequisite:

Applicants should have some fundamental programming exposure and experience before going to this course. Applicants ought to have an experience of fundamental development with any programming language, alongside user-level, working information on Mac, Unix/Linux, or Windows.

## Course Outline:

### Module 1:  An Overview of Python

- What is python?
- 1 -- 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

## 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 and 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
- The string module
- Reading CSV data

## Module 15:  Dates and times

- Working with dates and times
- Translating timestamps
- Parsing dates from text

## 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 fileinput
- Security and File Access

## Module 17: Network services

- Grabbing web content
- Detecting Malformed Input

## Module 18: Writing secure Python applications

- Parsing command-line options
- Getting help with pydoc
- Safely handling untrusted data
- Managing eval() permissions
- Potential insecure packages
- Embedding code snippets in Python
- Embedding authentication data in Python
- Potentially dangerous operations:
    - File access
    - Operating system access
    - Calls to external services
    - Called to external data sources
- Static analysis tools such as Bandit

## Module 19: Essential, Safe DB Access

- DB-API
- ORM with SQLAlchemy
- Preventing SQL Injection in DB Access
- Parameterization in DB Interactions

## Module 20: Log File Analysis

- Raw log file manipulation
- Fail2Ban
- Customizing Fail2Ban with Python

## Module 21: Security FIlters

- SQL-Injection Detection
- ModSecurity CRS filtering

## Module 22: Packet Analysis

- Packet Sniffing in Python

### Module 23: Analytics

- Security Logging and Analytics
- Attack Detection and Defense
- Python and Spark High-Level Overview