There are many good reasons to choose Python as your primary programming language. First of all Python is an easy to learn, powerful programming language. Furthermore, it has efficient high-level data structures, which allow you to write complex operations in fewer statements than in C, C++ or Java. Object-oriented programming is a lot easier than in languages like Java.

Python has become one of the most popular programming languages among developers and programmers. They praise it for its clean syntax and code readability. Python is a general-purpose high-level programming language. Python is both object oriented and imperative and it can be even used in a functional style as well. Python programs are portable, i.e. they can be ported to other operating systems like Windows, Linux, Unix and Mac OS X, and they can be run on Java and .NET virtual machines.

Course Duration: **5 + 3 days**

Objectives

This is an hands-on course, designed to help the developers to speed up in Python, as quickly as possible. The participant's in this course, will experience the following:

- Architecture
- Input and output
- Lists, Tuples & Dictionaries
- Decision Making & Loops
- Error handling.
- Function
- Garbage collection
- Object-oriented features. Classes & Objects
- Creating and using libraries and packages.
- Regular Expressions
- CGI (Common Gateway Interface) Programming
- Database access
- Network Programming
- Sending Email
- Multi-threaded Programming
- XML Processing
- GUI Programming

Lab requirements:

* As the training is highly lab oriented, each participant attending the training program must be provided with a Computer with the following software installed:

- Windows or Linux with Python 3.5

Audience
This course is designed for developers, system administrators, and test engineers, who wish to develop, automate, and test applications and systems using Python

Prerequisites
The participants should have prior programming experience and should be familiar with basic programming constructs.

Course Outline

[Day 1]

1: Python Introduction
 * what's Python?
 * Why do people use Python?
 * Some quotable quotes
 * A Python history lesson
 * Advocacy News
 * What's Python good for?
 * What's Python not good for?
 * The features list
 * Python portability
 * Summary

2. Using the Interpreter
 * Python's Interactive Prompt
 * Scripting
 * Program Execution Model
 * Program Architecture: modules
 * How to run Python programs
 * The IDLE interface
 * Other python IDEs

3. Python Scripting
 * Python Scripts in Linux/Unix & Windows
 * Whitespace Significance
 * Line Termination
 * Comments in Python
 * Basic Output Generation
 * Simple User Input
 * Python Modules
 * Module Search Paths
 * Determining the System Search Path
 * input()
 * raw_input()

4. Working with Variables in Python
 * Python Variables
 * Naming Conventions & Rules
 * Types as Objects
 * Variable References & Garbage Collection
 * Sequence Types
 * Membership Statements
 * List Iteration
 * Sequence Assignments
 * Mutable vs Immutable Objects
 * Multi Target Assignments

5. Numeric Operations in Python
 * More About Python's Numeric Types
 * Numeric Tools
 * The Decimal Module
 * Operator
 * Arithmetic
 * Logical
 * Relational
 * Bitwise
 * Special Operators
 * Operator Precedence

6. Python Compound Statements
 * Python Nesting Recap
 * Comparison Operations
 * The if Statement
 * The if Ternary Expression
 * The while Loop
 * The for Loop
 * Traversing Parallel Sets

7. Classes and Objects

* Introduction to OOP using python
* Classes and class attributes
* Instances and instance attributes
* Binding and method invocation
* Composition, Subclassing and Derivation
* Inheritance
* Built-in functions for classes, instances and other objects
* Privacy and Delegation
* An overview of built-in python classes and modules

[Day 2]

8. Python String Types

* Generating Strings in Python
* Immutable
* Common String Methods
* Type Conversion in Python
* Formatting String Output
* Format Specifier
* Variable Substitution
* String Indexing
* String Slicing
* String Iteration

9. Python's Tuples

* Immutable
* Common Tuples Methods
* Tuples Operations
* Tuples Indexing
* Tuples Slicing
* Tuples Iteration
* Multi-Dimensional Tuples (Matrices)

10. Python's Lists

* Common List Methods
* The range() Function
* List Operations
* String Indexing
* String Slicing
* String Iteration
* Multi-Dimensional Lists (Matrices)

11. Python List Comprehension

* Basic List Comprehensions
* Compound List Comprehensions

12. Python Dictionaries

* Python Dictionaries
* Assigning Values to Dictionaries
* Dictionary Methods
* Dictionaries vs Lists & Tuples
* Dictionary Indexing
* Dictionary Iteration

13. Basic Input/Output with Files

* Opening Files
* Working with Files
* Controlling Output Location

14. Handling Compound Data Structure

* Nested Structure
* Parsing and Loading into Data Structure
* Retrival of data from the Data Structure

15. Creating Python Functions

* Function Basics
* Defining Functions
* Function Polymorphism
* Argument Defaults
* Lambdas
* Local Variables
* Understanding __builtin__
* Preventing Variable Modifications
* Argument Matching Methods
* Keyword Argument Methods

[Day 3]

16. Regular Expression in Python

* Meta Characters
* re module
* Search
* Match
* Split
* Translation

17. Modules & Packages

* Module Basics
* Packages
* Using __all__ and _ Variables
* Using __name__

* Using third party modules

18. Exceptions
* About Exceptions
* Python's Default Exception Handler
* Using Try/Except/Else/Finally Exceptions
* Generating User Defined Exceptions
* More on Exceptions
* Exception Examples
* Using Asserts
* Exception Classes

[Day 4]

18. Common Gateway Interface (CGI) Programming
* CGI Architecture
* Writing CGI program
* HTTP Header
* CGI Environment Variables
* GET & POST Methods
* Using Cookies in CGI

19. MySQL Database access
* Installing PyMySQL
* Database Connection
* Creating Database Table
* INSERT Operation
* READ Operation
* Update Operation
* DELETE Operation
* COMMIT Operation
* ROLLBACK Operation

20. Network Programming
* What is Socket ?
* The socket module
* Server Socket Methods
* Client Socket Methods
* Developing Server & Client scripts
* Python Internet module

21. Sending Email
* Sending an HTML e-mail using Python
* Sending attachments as an E-mail

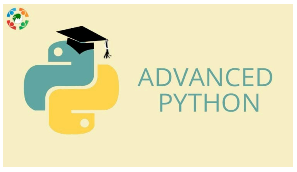
[Day 5]

22. Multi-threaded Programming
* Starting a New Thread
* The Threading Module
* Creating Thread Using Threading Module
* Synchronizing Threads
* Multithreaded Priority Queue

23. XML Processing
* XML Parser Architectures and APIs
* Parsing XML with SAZ APIs
* Parsing XML with DOM APIs

24. GUI Programming
* Tkinter Programming
* Tkinter Widgets
* Standard attributes
* Geometry Management

Python for Security Experts & Penetration Testing



***Python Scripting Expert (PSE &PT)** aims to teach you how to apply the powerful Python language to security research, penetration testing and attack automation using a fully hands-on practical approach with a gradual learning curve. This course has something for everyone – from the novice to the expert user!*

Module 1 of the course is meant for students who have very little or no programming experience before. We go really slowly on the basics of the language so you can pick up the essential programming skills before venturing to the more difficult modules. Module 2 onwards, we concentrate on application of Python programming to the field of computer security. If you are already a proficient Python programmer, you could start directly from Module 2.

Module 1: Python – Basic Fundamental

- Introduction to Interpreted Languages and Python
- Data Types and variables
- Operators and Expressions
- Statement Documentations and help
- String Revisited
- Control Flow
- Data structures in Python
- Functions ,Functional Programming
- File Handling
- Accessing the network and Internet
- Exception Handling
- Modular Programming
- Regular Expressions in Python
- OOPS concepts and Object Oriented Programming
- Modules, Packages and Distribution
- Configure Python in Linux and Unix
- Configure Python in Windows
- How to use Python in Mobiles: iPhone and Androids
- Python in Embedded Devices: Routers
- Python Program Portability
- Python Framework and Ides

Module 2: Managing files , directory and Security

- Input/output file system in Python
- Creating Managing File and Directory Access
- use of Multithreading and Concurrency
- how ,what and why Inter Process Communication (IPC)
- how to set Permissions and Controls.

Module 3: Network Security Programming – Sniffers and Packet Injectors

- Networking in Python
- An Introduction to Raw Socket basics
- Python Database access
- Porting Python code
- Socket Programming with Python
- Programming Servers and Clients
- Programming Wired and Wireless Sniffers
  - Packet Analyser
  - Packet Analyzer – Writing a Packet Sniffer in Python
  - Packet Analyzer (part 2) – Readability in Strings
  - Packet Analyzer (part 3) – Sequence in Acknowledgement Numbers in TCP
- Creating packet injector
- PCAP file parsing and analysis

Module 4: Web Application Security

- Introduction to web server and Application server
- Client Side scripting
- Intro to web application and penetration testing in web application
- HTML and XML file analysis
- Attacking Web Services and Countermeasure
- how and why to use Application Proxies and Data Mangling
- Automation of attacks such as SQL Injection, XSS etc.
- Intro to Buffer overflow and CSRF
- Web Application Fuzzers

Module 5: Exploitation Writing and Analysis Automation

- Exploit Development techniques
- Immunity Debuggers and Libs
- creating plugins in Python
- Binary data analysis
- Exploit analysis Automation

Module 6: Malware Analysis and Reverse Engineering

- Basics of Process Debugging
- Pydbg and its applications
- Analyzing live applications
- Setting breakpoints, reading memory etc.
- In-memory modifications and patching

Module 7: Setting Attack Task Automation

- Task Automation with Python
- Libraries and Applications

Module 8 - Gathering Information

- Info Gather (part 1) – An Activity in Post Exploitation Hacking
- Info Gather (part 2) – Enumerating Keys
- Info Gather (part 3) – Testing Python Scripts

* * * * * * * *