**Core Python**

**Duration: 5 days**

**Course Description:**

This training program provides a basic introduction to Python programming language. The focus of this training program will be more on **"PYTHONIC"** approach towards problem-solving, focused towards Application Development, Web App development and ML/AI readiness

**Target Audience (who should attend):**

\* Engineers who wish to learn python program to develop applications or to automate their applications/framework.

\* Engineers who wish to prototype new applications.

**Pre-requisites:**

\* Participants should be comfortable with the following technologies:

- Basic programming background with good understanding of programming

language ingredients that include variables and data types, flow control

statements, and function/procedural programming paradigms.

- Knowledge of any scripting language would be beneficial.

- Knowledge of OOP and modular programming concepts in any

programming language are recommended.

**Course Objectives:**

\* Understand the basic programming paradigms using Python 2 and Python 3

\* Understand built in data types, variables, functions and flow control

statements.

\* Learn how to use string, tuple, list, byte-array, set, dictionary types effectively.

\* Learn "pythonic" idioms and anti-idioms.

\* Learn functions, modules and file I/O operations.

\* Learn the basic concepts of OOP in Python.

\* Process, network, automation

\* Web scraping and web formats – json, html, xml

\* REST API development using Flask & consuming them

\* Threads, process, & co-routines

**Training mode:**

\* The Training program will be mostly demonstration oriented.

\* Most concepts will be taught by demonstrating code and participants are

expected to learn by practicing the same.

**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/Linux/Mac OSX with Python 3.10+ installed.

- A good programmer's editor (Notepad++ or PSPad on windows,

vim/emacs on Linux) installed.

\* LCD Projector with support to connect to trainer's laptop

\* Whiteboard, markers, scribbles pad + pen for participants.

**Course Agenda**

|  |  |
| --- | --- |
| **Day** | **Topics** |
| **Day1** | **Basics of Git**  **Introduction to Python**  **Getting started with Python**  **Working with Strings**  **Iterator Protocol** |
| **Day2** | **Lists, Tuples and Sets**  **Working with dictionaries**  **File Handling** |
| **Day3** | **Functions**  **Modular development**  **Classes and Objects**  **Errors and Exception handling**  **Standard Python Modules**  **Python Tools** |
| **Day4** | **Network Automation**  **Process Automation**  **Web Automation** |
| **Day5** | **Multi-Threading**  **Multi-Processing**  **Asynchronous**  **Project** |

**Day 1**

**Basics of Git**

\* Basics

\* Branching & merging

\* Synchronize

\* Make a change

**Concepts**

\* Object Based Vs Object Oriented

\* References

\* Memory Layout

\* Book keeping – packages/namespaces

\* Shallow copy / Deep Copy

\* Deep Dive into Objects in Memory

\* Garbage Collector

**Introduction to Python**

\* Introduction to python programming

\* Python implementations (features and differences):

- CPython, Jython, IronPython, PyPy, Stackless Python

\* Python 2 vs Python 3 (feature differences)

**Getting started with Python**

\* Numbers and expressions

\* Variables and statements

\* Conditional statements and loop

\* Handling user input

\* An overview of built-in functions and modules

\* Python syntax, style and coding conventions

\* Basic introspection using type() and dir() function

**Working with Strings**

\* An overview of strings in python

\* String operators

\* Built-in string manipulation functions

\* Built-in string methods

\* Special string features in python

\* Built-in modules for string handling

\* Unicode strings and bytearray

**Iterator Protocol (foreach loop)**

\* Iterable, Iterator, StopIteration, next

\* different types of iterators

**Day 2**

**Lists, Tuples, Sets & Dictionary**

\* Common sequence operations

\* Manipulation of Lists

\* Manipulation of Tuples

\* Manipulation of Sets

\* Manipulation of Dict

**Nested and Extended Data Structures**

\* Introduction to Nested Structures

**File Handling**

\* open, read, write, other operations

**Functions**

\* Creating user-defined functions

\* Passing functions

\* Formal arguments

\* Variable-length arguments

\* A walk-through on various built-in functions

**Day 3**

**Modular development – Modules & packages**

\* Creating modules

\* Variable scope

\* Understanding namespaces

\* Importing modules and module attributes

\* Module hierarchy

**Classes and Objects**

\* Introduction to OOP using python

\* Classes and class attributes

\* Instances and instance attributes

\* Binding and method invocation

\* Composition, Sub classing and Derivation

\* Inheritance

\* Built-in functions for classes, instances and other objects

\* An overview of built-in python classes and modules

**Errors and Exception Handling**

\* Introduction to exceptions

\* Detecting and handling exceptions

\* Exceptions as Strings and Classes

\* Raising exceptions

\* Creating user defined exceptions

\* Standard exceptions

**Standard Python modules (quick overview)**

\* Python Decorators

\* Generators

\* Using the sys and os module

\* Using the object persistence modules

\* Using the time and datetime modules

\* Using file related modules and logging modules

\* Using re module, numpy and pandas

**Python Tools**

\* unit testing – unittest & pytest

\* Debugger

\* Py Doc

\* Installing the PYTHON MODULEs

\* Distributing Python modules

\* virtual env

\* static analysis - pylint

**Day 4**

**Network Protocol Automation**

\* telnet, ftp, ssh, scp and smtp

**Process Automation**

\* subprocess & os module

**Working with JSON & Web Scrapping**

\* working with json parser – dump/dumps – load/loads

\* bs4 html parsing & xml parsing

\* requests modules – post/put/get/delete

\* Web scrapping – news-site, e-com site and others

\* REST calls with POSTMAN - requests

**Day 5**

**Mult-Threading**

\* Creating a Thread

\* Designing the thread functions & passing arguments, return

\* Thread Synchronization – mutex & condition

\* Thread communication

\* Thread Pools

**Mult-Processing**

\* Creating a Process & assign call backs

\* Designing the process functions & passing arguments, return

\* Process Synchronization – mutex, condition, semaphores

\* Inter Process Communication – list, dict, array, queue, value & pipe

\* Process Pools

**Asyncio**

\* Background of asyncio – pros & cons

\* Event Loop

\* Co-routine & Tasks

\* Futures & Synchronization

**Project -**