COMPLETE PYTHON Training

Course Duration: 2 months, Each Class: 1 hr, Weekly 6 days

Trainer: D. Chaitanya (IIT Roorkee Alumni), 20+ yrs Exp.

I. Core Python

1. Demo:
    a. Why Python?
    b. Who Uses Python today
    c. What can we do with Python
    d. How Python Developed and Supported
    e. Python – Technical Strengths.

2. Python Interpreter
3. Program Execution – programmer's view, Python's view
4. Installation
    a. Python
    b. All Related Software: PyCharm, VS Code, Jupyter Notebook
    c. Setup, configure Python in Laptop
5. All Numeric types in Python, *coding/Hands-on*
6. Python Variables, objects, References, Shared References, *coding/Hands-on*
7. Garbage Collection of objects
8. Strings in Python
    a. Basics
    b. String Literals - *coding/Hands-on*
    c. Indexing, Slicing - *coding/Hands-on*
    d. Methods
    e. Strings Module
    f. Formatting Expressions - *coding/Hands-on*
    g. Formatting method calls - *coding/Hands-on*

9. Lists in Python
    a. Basics
    b. Iterations and comprehensions - *coding/Hands-on*
    c. Indexing, Slicing and Matrixes - *coding/Hands-on*
    d. Methods

10. Dictionaries
    a. Basics

b. Different ways to create Dictionary - *coding/Hands-on*
c. Methods

11. Tuples, Sets
    a. Basics - *coding/Hands-on*
    b. Difference between List & Tuples, Lists & Sets
    c. Named Tuples - *coding/Hands-on*
    d. Methods

12. Files
    a. Opening & Using Files - *coding/Hands-on*
    b. Working with Files - *coding/Hands-on*
    c. Pickling & Unpickling - *coding/Hands-on*
    d. File context managers - *coding/Hands-on*

13. Python Statements - *coding/Hands-on*
    a. Assignments, Expressions and Prints
    b. **if**-**else**, **if**-**elif**-**else**, **if**-**else** ternary expression
    c. **while** and **for** loops
    d. Comprehensions vs regular
    e. Parallel Traversals: **map** and **zip** functions
    f. Other important functions: range, len, enumerate

14. Iterations and Comprehensions - *coding/Hands-on*
15. Python online Documentation
16. Python Functions - *coding/Hands-on*
    a. **def** statements
    b. definitions and calls
    c. variables

17. Variable Scopes
    a. Basics
    b. LEGB rule
    c. Nested Functions - *coding/Hands-on*
    d. **global**, **nonlocal** statements - *coding/Hands-on*

18. Function Arguments - *coding/Hands-on*
    a. Arguments and shared references
    b. Arguments passing basics
    c. Arguments matching basics

   d. Arguments matching syntax

19. Advanced Function Concepts - *coding/Hands-on*
   a. Function objects: Attributes
   b. Anonymous Functions: **lambda**, basics, nested lambda

20. Generators and comprehensions - *coding/Hands-on*
   a. Generator functions
   b. yield vs return
   c. Generator functions vs Generator Expressions

21. Python Modules
   a. Definition, why modules?
   b. Typical Python program architecture
   c. Import statement - *coding/Hands-on*
   d. How Import works: Find it, Compile it, Run it
   e. Standard library modules
   f. **__pycache__** folder for byte code files
   g. Module search path

22. Module coding Basics
   a. Module creation
   b. **import** statement, **from** statement, **from *** statement
   c. Module Namespaces, Namespace dictionaries: **__dict__**

23. Module Packages
   a. Package import basics
   b. Why Package imports?
   c. Relative import basics
   d. Why relative imports?
   e. Package Namespaces

24. Advanced Module Topics
   a. Mixed usage modes: __name__ and __main__ - *coding/Hands-on*
   b. The **as** extension for **import** and **from -** *coding/Hands-on*

25. Introduction to Python Classes - *coding/Hands-on*
   a. Why Classes?
   b. Classes, constructors and Instances
   c. Method calls
   d. Attribute inheritance search

e. OOP is about code reuse
f. Subclassing by Inheritance
g. Polymorphism in Action
h. Class vs instance attributes

26. Coding with Classes - *coding/Hands-on*
    a. Abstract super classes
    b. Nested classes

27. Operator Overloading - *coding/Hands-on*
    a. Constructors: __init__
    b. Indexing, Slicing: __getitem__ and __setitem__
    c. Attribute Access: __getattr__ and __setattr__
    d. String Representation: __repr__ and __str__
    e. Right side and In-Place Uses: __radd__ and __iadd__
    f. Call Expressions: __call__
    g. Comparisons: __lt__, __gt__ and others
    h. Boolean Tests: __bool__ and __len__
    i. Destructors: __del__

28. Special features of Classes - *coding/Hands-on*
    a. Inheritance: “IS-a” relationship
    b. Composition: “HAS-a” relationship
    c. Class objects

29. Advanced Class Topics - *coding/Hands-on*
    a. “New style” class model
    b. Diamond inheritance change
    c. MRO: Method Resolution Order
    d. Static and Class methods
    e. The “**super**” built-in function

30. Exception Basics - *coding/Hands-on*
    a. Why Exceptions?
    b. Default Exception handler
    c. Catching Exceptions
    d. Raising Exceptions

31. Coding Exceptions - *coding/Hands-on*
    a. The try/except/else statement

b. try/finally statement
c. raise statement
d. assert statement
e. with/as context managers
f. Nesting Exception Handlers

32. Exception Objects
    a. Class based exceptions
    b. Why Exception hierarchies?
    c. Built-in Exception Classes
    d. Custom Exceptions

II. Regular Expressions with Python:

1. Why regular expressions
2. What is regular expression?
3. **re** module
4. compiling regular expression
5. commonly used functions in **re** module
6. meta characters in regular expressions
7. Basic patterns in regular expressions
8. Advanced patterns in regular expressions

III. Data Libraries with Python:

1. Introduction to numpy
2. Creating arrays
3. Indexing Arrays
4. Array Transposition
5. Universal Array Function
6. Array Processing
7. Array Input and Output
8. Introduction to Pandas
9. Data reading with Pandas
10. Data cleaning with Pandas
11. Data wrangling with Pandas
12. Data selection with Pandas
13. Data extraction with Pandas