Futurepoint
SHAPING U'R FUTURE

Python Course Content

Course Description:

This course teaches students how to develop Python applications. Topics covered include the Python programming language syntax, OO programming using Python, exception handling, file input/output, Database connectivity.

Who Should Attend:

This course is designed for application programmers and designers planning to develop applications using the Python.

Benefits of Attendance:

Upon completion of this course, students will be able to:

Run a Python application.

Navigate through the API docs.

Use the Object Oriented paradigm in Python

Understand the division of classes into Python packages.

Use Exceptions to handle run time errors.

Prerequisites:

Knowledge of programing languages such as C or C++ is desirable but not mandatory.

Python Course Content

Core Python

Introduction to Script

- What is Script, program?
- Types of Scripts
- Difference between Script and Programming Languages
- Features and Limitation of Scripting
- Types of programming Language Paradigms

Introduction to Python

- What is Python?
- Why Python?
- Who Uses Python?
- Characteristics of Python

- History of Python
- What is PSF?
- Python Versions
- How to Download and Install Python
- Install Python with Diff IDEs
- Features and Limitations of Python
- Python Applications
- Creating Your First Python Program
- Printing to the Screen
- Reading Keyboard Input
- Using Command Prompt and GUI or IDE
- Python Distributions

Different Modes in PYTHON

- Execute the Script
- Interactive and Script Mode
- Python File Extensions
- SETTING PATH IN Windows
- Clear screen inside python
- Learn Python Main Function
- Python Comments
- Quit the Python Shell
- Shell as a Simple Calculator
- Order of operations
- Multiline Statements
- Quotations in Python
- Python Path Testing
- Joining two lines
- Python Implementation Alternatives
- Sub Packages in Python
- Uses of Python in Data Science, IoT
- Working with Python in Unix/Linux/Windows/Mac/Android..!!

PYTHON NEW IDEs

- PyCharm IDE
- How to Work on PyCharm
- PyCharm Components
- Debugging process in PyCharm
- PYTHON Install Anaconda

- What is Anaconda?
- Coding Environments
- Spyder Components
- General Spyder Features
- Spyder Shortcut Keys
- Jupyter Notebook
- What is Conda? and Conda List?
- Jupyter and Kernels
- What is PIP?

Variables in Python

- What is Variable?
- Variables and Constants in Python
- Variable, Variable names and Value
- Mnemonic Variable Names
- Values and Types
- What Does “Type” Mean?
- Multiple Assignment
- Python different numerical types
- Standard Data Types
- Operators and Operands
- Order of Operations
- Swap variables
- Python Mathematics
- Type Conversion
- Mutable Versus Immutable Objects

String Handling

- What is string?
- String operations and indices
- Basic String Operations
- String Functions, Methods
- Delete a string
- String Multiplication and concatenation
- Python Keywords, Identifiers and Literals
- String Formatting Operator
- Structuring with indentation in Python
- Built-in String Methods
- Define Data Structure?

- Data Structures in PYTHON

Python Operators and Operands

- Arithmetic, Relational Operators and Comparison Operators
- Python Assignment Operators
- Short hand Assignment Operators
- Logical Operators or Bitwise Operators
- Membership Operators
- Identity Operators
- Operator precedence
- Evaluating Expressions

Python Conditional Statements

- How to use “if condition” in conditional structures
- if statement (One-Way Decisions)
- if .. else statement (Two-way Decisions)
- How to use “else condition”
- if .. elif .. else statement (Multi-way)
- When “else condition” does not work
- How to use “elif” condition
- How to execute conditional statement with minimal code
- Nested IF Statement

Python LOOPS

- How to use “While Loop” and “For Loop”
- How to use For Loop for set of other things besides numbers
- Break statements, Continue statement, Enumerate function for For Loop
- Practical Example
- How to use for loop to repeat the same statement over and again
- Break, continue statements

Learning Python Strings

- Accessing Values in Strings
- Various String Operators
- Some more examples
- Python String replace() Method

- Changing upper and lower case strings
- Using “join” function for the string
- Reversing String
- Split Strings

Sequence or Collections in PYTHON

- Strings
- Unicode Strings
- Lists
- Tuples
- buffers
- xrange

Python Lists

- Lists are mutable
- Getting to Lists
- List indices
- Traversing a list
- List operations, slices and methods
- Map, filter and reduce
- Deleting elements
- Lists and strings

Python TUPLE

- Advantages of Tuple over List
- Packing and Unpacking
- Comparing tuples
- Creating nested tuple
- Using tuples as keys in dictionaries
- Deleting Tuples
- Slicing of Tuple
- Tuple Membership Test
- Built-in functions with Tuple
- Dotted Charts

Python Sets

- How to create a set?

- Iteration Over Sets
- Python Set Methods
- Python Set Operations
- Union of sets
- Built-in Functions with Set
- Python Frozenset

Python Dictionary

- How to create a dictionary?
- PYTHON HASHING?
- Python Dictionary Methods
- Copying dictionary
- Updating Dictionary
- Delete Keys from the dictionary
- Dictionary items() Method
- Sorting the Dictionary
- Python Dictionary in-built Functions
- Dictionary len() Method
- Variable Types
- Python List cmp() Method
- Dictionary Str(dict)

Python Functions

- What is a function?
- How to define and call a function in Python
- Types of Functions
- Significance of Indentation (Space) in Python
- How Function Return Value?
- Types of Arguments in Functions
- Default Arguments and Non-Default Arguments
- Keyword Argument and Non-keyword Arguments
- Arbitrary Arguments
- Rules to define a function in Python
- Various Forms of Function Arguments
- Scope and Lifetime of variables
- Nested Functions
- Call By Value, Call by Reference
- Anonymous Functions/Lambda functions

- Passing functions to function
- map(), filter(), reduce() functions
- What is a Docstring?

Advanced Python

Python Modules

- What is a Module?
- Types of Modules
- The import Statement
- The from…import Statement
- ..import * Statement
- Underscores in Python
- The dir( ) Function
- Creating User defined Modules
- Command line Arguments
- Python Module Search Path

Packages in Python

- What is a Package?
- Introduction to Packages?
- py file
- Importing module from a package
- Creating a Package
- Creating Sub Package
- Importing from Sub-Packages
- Popular Python Packages

Python Date and Time

- How to Use Date & DateTime Class
- How to Format Time Output
- How to use Timedelta Objects
- Calendar in Python
- datetime classes in Python
- How to Format Time Output?
- The Time Module
- Python Calendar Module
- Python Text Calendar, HTML Calendar Class

- Unix Date and Time Commands

File Handling

- What is a data, Information File?
- File Objects
- File Different Modes and Object Attributes
- How to create a Text Fil and Append Data to a File and Read a File
- Closing a file
- Read, read line ,read lines, write, write lines…!!
- Renaming and Deleting Files
- Directories in Python
- Working with CSV files and CSV Module
- Handling IO Exceptions

Python OS Module

- Shell Script Commands
- Various OS operations in Python
- Python File System Shell Methods

Python Exception Handling

- Python Errors
- Common RunTime Errors in PYTHON
- Abnormal termination
- Chain of importance Of Exception
- Exception Handling
- Try … Except
- Try .. Except .. else
- Try … finally
- Argument of an Exception
- Python Custom Exceptions
- Ignore Errors
- Assertions
- UsingAssertionsEffectively

More Advanced PYTHON

- Python Iterators, Generators, Closures, Decorators and Python @property

Python Class and Objects

- Introduction to OOPs Programming
- Object Oriented Programming System
- OOPS Principles
- Define Classes
- Creating Objects
- Class variables and Instance Variables Constructors
- Basic concept of Object and Classes
- Access Modifiers
- How to define Python classes
- Python Namespace
- Self-variable in python
- Garbage Collection
- What is Inheritance? Types of Inheritance?
- How Inheritance works?
- Python Multiple Inheritance
- Overloading and Over Riding
- Polymorphism
- Abstraction
- Encapsulation
- Built-In Class Attributes

Python Regular Expressions

- What is Regular Expression?
- Regular Expression Syntax
- Understanding Regular Expressions
- Regular Expression Patterns
- Literal characters
- Repetition Cases
- Example of w+ and ^ Expression
- Example of \s expression in re.split function
- Using regular expression methods
- Using re.match()
- Finding Pattern in Text (re.search())

- Using re.findall for text
- Python Flags
- Methods of Regular Expressions

Python XML Parser

- What is XML?
- Difference between XML and HTML and XML, JSON, Gson
- How to Parse XML and Create XML Node
- Python vs JAVA
- XML and HTML

Python-Data Base Communication

- What is Database? Types of Databases?
- What is DBMS?, RDBMS?
- What is Big Data? Types of data?
- Oracle
- MySQL
- SQL server
- DB2
- Postgre SQL
- Executing the Queries
- Bind Variables
- Installing of Oracle Python Modules
- Executing DML Operations..!!

Multi-Threading

- What is Multi-Threading
- Threading Module
- Defining a Thread
- Thread Synchronization

GUI Programming-Tkinter

- Introduction
- Components and Events
- Adding Controls
- Entry Widget, Text Widget, Radio Button, Check Button
- List Boxes, Menus, ComboBox