Introduction to Python
Course Outline

Introduction:

This Introduction to Python course provides a comprehensive foundation in Python programming, starting with Python basics such as running Python scripts, understanding literals, comments, data types, and working with variables. The course then progresses to functions and modules, arithmetic operations, and Python string manipulation. You'll also learn about iterables, including sequences, dictionaries, and sets, as well as virtual environments, packages, and pip. The course covers flow control, exception handling, Python dates and times, and file processing. Finally, it introduces you to PEP8 and Pylint for code quality and style. Each section includes exercises to help you practice your newfound skills and solidify your understanding of Python programming concepts.

Course Duration:

4 days

Prerequisites:

There are no prerequisites but some programming experience is helpful.

Audience Profile:

Anyone interested in learning about Python.

Benefits:

1. **Easy-to-follow curriculum**: The course is structured to gradually introduce Python concepts, starting with the basics and progressing to more advanced topics. This step-by-step approach allows participants to build their skills and confidence at a comfortable pace.
2. **Engaging, hands-on learning**: Our course emphasizes practical coding exercises and real-life examples, enabling students to apply their newfound Python knowledge to actual programming challenges. This hands-on approach helps solidify understanding and retention.
3. **Strong foundation for future learning**: By covering fundamental Python concepts, the course lays the groundwork for participants to pursue more advanced programming topics, setting them on a path to becoming proficient Python developers.

Outline:

1. **Python Basics**

   This module explains how to get started using Python.

   - Getting Familiar with the Terminal
   - Running Python
   - Running a Python File
   - Hello, world! (Exercise)
   - Literals, Comments, and Data Types
   - Exploring Types (Exercise)
   - Variables

Introduction to Python
Course Outline

- A Simple Python Script (Exercise)
- Constants and Deleting Variables
- Writing a Python Module
- `print()` Function
- Collecting User Input
- Hello, You! (Exercise)
- Reading from and Writing to Files
- Working with Files (Exercise)

After completing this module, students will be able to:

- work with Python, use variables, output data, collect user input, and write simple Python functions and modules.

2. Functions and Modules

You have seen some of Python's built-in functions. In this lesson, you will learn to write your own.

- Defining Functions
- Variable Scope
- Global Variables
- Function Parameters
- A Function with Parameters (Exercise)
- Default Values
- Parameters with Default Values (Exercise)
- Returning Values
- Importing Modules
- Methods vs. Functions

After completing this module, students will be able to:

- define functions with or without parameters
- understand variable scope and how to import modules.

3. Math

Python includes some built-in math functions and some additional built-in libraries that provide extended math (and related) functionality. In this lesson, we'll cover the built-in functions and the math and random libraries.

- Arithmetic Operators
- Floor and Modulus (Exercise)
- Assignment Operators
- Precedence of Operations
- Built-in Math Functions
- The `math` Module
- The `random` Module
- How Many Pizzas Do We Need? (Exercise)

- Dice Rolling (Exercise)

After completing this module, students will be able to:

- do basic math in Python
- use the math and random modules for extended math functionality.

4. Python Strings

This module explains how to work with Python Strings.

- Quotation Marks and Special Characters
- String Indexing
- Indexing Strings (Exercise)
- Slicing Strings
- Slicing Strings (Exercise)
- Concatenation and Repetition
- Repetition (Exercise)
- Combining Concatenation and Repetition
- Python Strings are Immutable
- Common String Methods
- String Formatting
- Playing with Formatting (Exercise)
- Formatted String Literals (f-strings)
- Built-in String Functions
- Outputting Tab-delimited Text (Exercise)

After completing this module, students will be able to:

- manipulate and format strings.

5. Iterables: Sequences, Dictionaries, and Sets

Iterables are objects that can return their members one at a time. The iterables we will cover in this module are lists, tuples, ranges, dictionaries, and sets.

- Definitions
- Sequences
- Lists
- Sequences and Random
- Remove and Return Random Element (Exercise)
- Tuples
- Ranges
- Converting Sequences to Lists
- Indexing
- Simple Rock, Paper, Scissors Game (Exercise)
- Slicing
- Slicing Sequences (Exercise)
- min(), max(), and sum()

Introduction to Python
Course Outline

- Converting between Sequences and Strings
- Unpacking Sequences
- Dictionaries
- The len() Function
- Creating a Dictionary from User Input (Exercise)
- Sets
- *args and **kwargs

After completing this module, students will be able to:

- understand lists, tuples, ranges, dictionaries, and sets.
- understand the *args and **kwargs parameters.

6. Virtual Environments, Packages, and pip

A virtual environment provides a self-contained directory tree with its own Python installation and additional packages necessary for the project(s) being done in that environment. As such, scripts can be run in a virtual environment that have dependencies that are different from those in other development projects that may be running in the standard environment or in separate virtual environments.

- Creating, Activating, Deactivating, and Deleting a Virtual Environment (Exercise)
- Packages with pip
- Working with a Virtual Environment (Exercise)

After completing this module, students will be able to:

- create and use virtual environments
- install packages with pip.

7. Flow Control

This module explains how to change the flow by using conditional statements and loops.

- Conditional Statements
- Compound Conditions
- The is and is not Operators
- all() and any() and the Ternary Operator
- In Between
- Loops in Python
- All True and Any True (Exercise)
- break and continue
- Looping through Lines in a File
- Word Guessing Game (Exercise)
- The else Clause in Loops
- for...else (Exercise)
- The enumerate() Function
- Generators
- List Comprehensions

Introduction to Python
Course Outline

After completing this module, students will be able to:

- write if-elif-else conditions and to loop through sequences.
- understand the enumerate() function, generators, and list comprehensions.

8. Exception Handling

This module explains how to anticipate and handle exceptions gracefully.

- Exception Basics
- Generic Exceptions
- Raising Exceptions (Exercise)
- The `else` and `finally` Clauses
- Using Exceptions for Flow Control
- Running Sum (Exercise)
- Raising Your Own Exceptions

After completing this module, students will be able to:

- handle Python exceptions.

9. Python Dates and Times

This module explains how to use Python's built-in modules to work with dates and times.

- Understanding Time
- The time Module
- Time Structures
- Times as Strings
- Time and Formatted Strings
- Pausing Execution with `time.sleep()`
- The `datetime` Module
- `datetime.datetime` Objects
- What Color Pants Should I Wear? (Exercise)
- `datetime.timedelta` Objects
- Report on Departure Times (Exercise)

After completing this module, students will be able to:

- work with the time and datetime modules

10. File Processing

This module explains how to process files.

- Paths
- The `pathlib` Module
- Opening Files
- Finding Text in a File (Exercise)

- Writing to Files
- Writing to Files (Exercise)
- List Creator (Exercise)
- Path Methods for Reading and Writing Files
- Making Directories
- Deleting Files and Directories
- Renaming Files and Directories
- The `os` Module
- A Better Way to Open Files
- Comparing Lists (Exercise)

After completing this module, students will be able to:

- work with files and directories on the operating system

11. PEP8 and Pylint

This module explains how to use the official Python style guide.

- PEP8
- Pylint

After completing this module, students will be able to:

- understand the PEP8 coding standards and how to use Pylint to analyze your code.