# UQ24CA651A - MCA (4-0-2-5-5)

## Course Content:

### *Unit 1: Python Basics and Data Structures*

Introduction to Python - History of Python, Introduction to Python - Advantages of using Python, Introduction to Python - Review of Data types, Functions - Defining functions, Functions - Parameters and arguments, Functions - Return statement, Functions - Scope of variables Random package, Comprehensions - Syntax of comprehensions, Comprehensions - Examples of comprehensions, Iterators and Decorators - Creating iterators and decorators, Iterators and Decorators - Examples of iterators and decorators

### *Experiential Learning:*

Worksheets on Data Types, Control Structures, Implement Functions, comprehensions, decorators

### *No of Hours :*

1 4 + 7 H o u r s

### *Unit 2: Object-Oriented Programming (OOP) and Advanced Concepts*

Introduction to OOP - Explanation of OOP, Introduction to OOP - uses, Introduction to OOP - comparison with procedural programming, Classes, Objects, and Inheritance - Creating and handling classes (__init__), Classes, Objects, and Inheritance - objects, Classes, Objects, and Inheritance - static variables, Classes, Objects, and Inheritance - understanding inheritance and polymorphism, Encapsulation, Constructors, and Destructors - Understanding encapsulation, Encapsulation, Constructors, and Destructors - using constructors and destructors for object management, Advanced OOP Concepts - Delving into static and class methods, Advanced OOP Concepts - class variables, Composition and aggregation, Metaclasses, Introspection

### *Experiential Learning:*

Use hands-on examples to understand OOP concepts and compare them with procedural programming., Create classes and objects, practice using constructors and destructors., Apply inheritance and polymorphism by designing classes in a real-world model., Explore encapsulation by using private members., Finally, interact with advanced OOP concepts like static and class methods, static variables, and get to understand the power of composition and aggregation., Create classes dynamically.

### *No of Hours :*

1 4 + 7 H o u r s

### *Unit 3: Exception Handling, File I/O, Modules and Packages*

Error Handling - Introduction to errors and exceptions in Python, Error Handling - handling exceptions, Error Handling - raising exceptions, Error Handling - creating custom exceptions, File I/O Operations - Understanding reading and writing CSV, JSON, and XML files, File I/O Operations - object serialization with Pickle, Python Modules and Packages - Exploring modules, Python

Modules and Packages - import statements, Python Modules and Packages - packages, and directory structure, Python Standard Library - Overview of Python Standard Library, Python Standard Library - usage of datetime, os, sys and collections modules

### *Experiential Learning:*

Experience error handling firsthand by raising, catching, and handling exceptions in real scenarios., Learn to perform file operations with CSV, JSON, and XML files, and grasp object serialization with Pickle. Work with Python's modules and packages, exploring their structure and import mechanics.”,“Understand Python's standard library modules like datetime, os, sys and collections through practical usage.

### *No of Hours :*

1 4 + 7 H o u r s

### *Unit 4: Web Frameworks*

Introduction to Flask, Flask Basics, Flask application structure - Routes and view functions, Templates with Jinja2, Static files, Flask Forms and User Input - Flask-WTF, Creating and rendering forms, Form validation, Handling user input, Building a Simple Flask Application, Flask-SQLAlchemy and Databases - Configuring Flask-SQLAlchemy, Flask-SQLAlchemy and Databases - Creating models, Flask-SQLAlchemy and Databases - Basic CRUD operations, Deploying the Application on Cloud

### *Experiential Learning:*

Designing the application - Setting up models - views, and templates - Adding user input and form handling - Implementing basic functionality, Get exposed to knowledge of Flask-SQLAlchemy by setting up databases, creating models, and performing CRUD operations., Design, build, and implement a simple Flask application. Deploy their application (Flask) on PythonAnywhere.

### *No of Hours :*

1 4 + 7 H o u r s