FACULTY OF COMPUTER APPLICATIONS
Master of Computer Applications

- **Sem.** : 1
- **Subject Code** : 05MC0107
- **Subject** : Programming with Python & Problem Solving
- **Course Objectives** :

1. To understand the fundamental concepts of Python programming, including syntax, data types, control structures, and functions.
2. To study the usage of Python data structures such as strings, lists, tuples, sets, and dictionaries in problem solving.
3. To understand object-oriented programming concepts in Python including classes, inheritance, polymorphism, and abstraction.
4. To learn file handling, exception handling, and regular expressions for effective data processing and error management.
5. To analyze and apply Python libraries, data analysis tools, and database connectivity for real-world applications

- **Prerequisites** : Knowledge of any programming language

Unit No	Topics Covered	No of lectures required

FACULTY OF COMPUTER APPLICATIONS
Master of Computer Applications

1	**Python Programming Foundations, Data Structures and Functions** **Python Programming Basics** ● Introduction to Python: features, versions, and applications ● Installation and environment setup ● Anaconda, VS Code, Jupyter Notebook ● Python execution using IDLE and interactive shells ● Variables, dynamic typing, data types and type casting ● Input and output operations ● Expressions, assignment statements and operators **Data Structures in Python** **Strings** Slicing, formatting, built-in string functions **Lists** Operations, slicing, list comprehensions **Tuples and Sets** Dictionaries Operations, iteration and dictionary methods **Mutable and Immutable Objects** **Functions in Python** User-defined functions Arguments and return values **Types of arguments** Required, keyword, variable-length Lambda (anonymous) functions **Functional programming tools** map(), filter(), reduce() Introduction to recursion	**10**

2	**Control Structures, Iteration and Intermediate Python Concepts** **Conditional statements** if, if-else, if-elif-else Nested conditions **Looping constructs** for loop, while loop **Loop control statements** break, continue, pass **Intermediate Python Concepts** Iterators and Iterables Generator functions and yield keyword **Comprehensions** List, dictionary and set comprehensions **Variable scope** local, global and nonlocal variables	**8**

3	**Modules, Packages, Standard Libraries and Regular Expressions** **Modules and Packages** Creating and using modules **Import mechanisms** import module from module import Python package structure PYTHONPATH and virtual environments Using pip for package management **Python Standard Libraries** math, random, date time os, sys, shutil **Regular Expressions** Basics of regular expressions **Pattern matching using re module** match(), search(), find all(), compile(), sub() Sequence characters, quantifiers and special characters Using regular expressions to extract information from files	**8**

FACULTY OF COMPUTER APPLICATIONS
Master of Computer Applications

4	**Object-Oriented Programming and File Handling** **Object-Oriented Programming in Python** Classes and objects Constructor and destructor Use of self keyword Instance variables and class variables Instance methods, class methods and static methods **Inheritance** Single, multilevel and multiple inheritance Method overriding and polymorphism Method Resolution Order (MRO) Operator overloading and magic methods Encapsulation and abstraction Abstract classes and interfaces in Python **File Handling** **File modes** r, w, a, x, binary modes Reading and writing files Working with CSV files **File pointer operations** seek() and tell() Random access using binary files Zipping and unzipping files	**9**

5	**Exception Handling, Data Analysis and Database Connectivity** **Exception Handling and Debugging** Types of errors in Python Built-in exceptions **Exception handling blocks** try, except, else, finally Defining and raising exceptions Assertion using assert statement Logging errors and debugging strategies **Data Analysis and Visualization** Introduction to Pandas **Creating Data Frames from** CSV files, Excel files, lists and tuples **Data inspection and operations** head(), tail(), shape, columns, size, dtypes describe(), max(), min() **Indexing and sorting** loc, set_index(), reset_index(), sort_values() **Handling missing data** dropna(), fillna() **Drawing graphs** Line, Bar, Histogram and Pie charts **Database Connectivity with Python** Overview of database connectivity modules Installing and configuring MySQLdb Connecting to databases Cursor creation and usage Executing SQL commands Insert, update and delete operations **Fetching records** fetchone(), fetchmany(), fetchall() Using Python and regular expressions to retrieve database records	**10**

FACULTY OF COMPUTER APPLICATIONS
Master of Computer Applications

Course Outcomes:

1. Student will be able to design programs using logical thinking, control flow, and core programming concepts with appropriate data types, data structures, and file handling.

2. Student will be able to develop Python applications that store, process, and analyze data using loops, functions, exception handling, and basic Python libraries.

3. Student will be able to solve problems efficiently by applying reusable components, file operations, and decision-making, iteration, and functional concepts in Python.

4. Student will be able to create organized and maintainable Python programs using data structures, control mechanisms, modular functions, and exception-handling techniques.

5. Student will be able to work with files and standard Python libraries to perform basic numerical operations and data handling tasks.

Course Outcomes – Program Outcomes Mapping Table :

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11
CO1											
CO2											
CO3											
CO4											
CO5											

Text Book:

1. Core Python Programming, R. Nageswara Rao, Dreamtech Press, 2nd Edition.

Reference Books:

1. Introduction to Computation and Programming Using Python, John V. Guttag, Prentice Hall of India, Second Edition.
2. Core Python Applications Programming, Wesley J. Chun, Pearson, 3rd Edition.
3. Professional Python, Luke Sneeringer, WROX Publication, First Edition.

Web References:

1. Learn Python the Hard Way, http://learnpythonthehardway.org/
2. Swaroop C. H., A Byte of Python, http://www.swaroopch.com/notes/python
3. Dive Into Python 3, http://www.diveintopython.net/
4. Python for Beginners, Python.org, https://www.python.org/about/gettingstarted/

App References:

1. Learn Python
2. Solo Learn Python
3. Python Pattern Programs Free
4. Python Programming App

Syllabus Coverage from text /reference book & web/app reference:

Unit #	Chapter Numbers
1	1,2,9
2	6
3	18
4	12-15,17
5	24,25

PRACTICALS

Unit No	List of Practicals

FACULTY OF COMPUTER APPLICATIONS
Master of Computer Applications

1	1. Write a program to demonstrate basic input and output operations in Python. 2. Write a program to illustrate the use of different data types and type casting. 3. Write a program to perform arithmetic relational and logical operations using Python operators. 4. Write a program to demonstrate string operations including slicing formatting and built-in string functions. 5. Write a program to create and manipulate lists using indexing slicing and list comprehensions. 6. Write a program to illustrate the use of tuples and sets with basic operations. 7. Write a program to create a dictionary and demonstrate dictionary methods and iteration. 8. Write a program to explain mutable and immutable objects in Python. 9. Write a program to define and use user-defined functions with different types of arguments. 10. Write a program to demonstrate recursion using factorial or Fibonacci series.
2	1. Write a program to demonstrate conditional statements using if if-else and if-elif-else. 2. Write a program to check whether a number is positive negative or zero using nested conditions. 3. Write a program to generate a multiplication table using a for loop. 4. Write a program to find the sum of digits of a number using a while loop. 5. Write a program to demonstrate the use of break continue and pass statements. 6. Write a program to iterate over lists strings and dictionaries using loops. 7. Write a program to demonstrate list dictionary and set comprehensions. 8. Write a program to illustrate variable scope using local global and nonlocal variables. 9. Write a program to demonstrate iterators and iterables in Python. 10. Write a program to generate a sequence of numbers using generator functions and yield keyword.

3	1. Write a program to create and import a user-defined module. 2. Write a program to demonstrate different import mechanisms in Python. 3. Write a program to perform mathematical operations using math module. 4. Write a program to generate random numbers using random module. 5. Write a program to display current date and time using date time module. 6. Write a program to perform file and directory operations using os and sys modules. 7. Write a program to copy move and delete files using shut il module. 8. Write a program to demonstrate basic regular expression pattern matching. 9. Write a program to use re module functions such as match search and find all. 10. Write a program to extract specific information from a text file using regular expressions.
4	1. Write a program to create a class and object in Python. 2. Write a program to demonstrate constructor and destructor usage. 3. Write a program to illustrate instance variables and class variables. 4. Write a program to demonstrate instance methods class methods and static methods. 5. Write a program to implement single multilevel and multiple inheritance. 6. Write a program to demonstrate method overriding and polymorphism. 7. Write a program to illustrate method resolution order and magic methods. 8. Write a program to demonstrate encapsulation and abstraction using classes. 9. Write a program to read and write data into a file using different file modes. 10. Write a program to perform file pointer operations using seek and tell methods.

5	1. Write a program to demonstrate different types of errors and exceptions in Python. 2. Write a program to handle exceptions using try except else and finally blocks. 3. Write a program to define and raise user-defined exceptions. 4. Write a program to demonstrate the use of assert statement. 5. Write a program to perform basic data analysis using Pandas Data Frame. 6. Write a program to create a Data Frame from CSV or Excel files. 7. Write a program to handle missing data using drop na and fill na methods. 8. Write a program to plot line bar histogram and pie charts using Python. 9. Write a program to connect Python with a database and execute SQL queries. 10. Write a program to retrieve database records using fetch one fetch many and fetch all methods.