

Python Lab Programs

MCA-24-352

UNIT 1: Introduction to Python Programming (Basic Programs)

- Program to print “Hello, World!”.
- Program to take user input and display it.
- Program to swap two numbers (with and without third variable).
- Program to find the square and cube of a number.
- Program to perform basic arithmetic operations (+, −, ×, ÷).
- Program to calculate simple interest and compound interest.
- Program to convert temperature from Celsius to Fahrenheit.
- Program to check if a number is even or odd.
- Program to check if a number is positive, negative, or zero.
- Program to find the largest of two numbers.
- Program to find the largest among three numbers.
- Program to check whether a number is divisible by 5 and 11.
- Program to check leap year.
- Program to check whether a character is a vowel or consonant.
- Program to calculate the grade of a student based on marks.
- Program to check whether a given number is prime.
- Program to print the first N natural numbers.
- Program to print multiplication table of any number.
- Program to calculate the sum of first N natural numbers.
- Program to calculate the factorial of a number.
- Program to generate the Fibonacci series.
- Program to count the number of digits in a given integer.
- Program to reverse a number using a loop.
- Program to find the length of a string without using `len()`.
- Program to count vowels, consonants, digits, and special characters.
- Program to reverse a string.
- Program to check whether a string is palindrome.
- Program to count the frequency of each character in a string.
- Program to define a function to add two numbers.
- Program to define a function to find the factorial of a number.

UNIT 2: Collections in Python

A. List Programs

- Create a list and perform insertion, deletion, and update operations.
- Find the largest and smallest element in a list.

- Remove duplicate elements from a list.
- Count the frequency of each element in a list.
- Reverse a list without using built-in functions.
- Merge two lists into a single list.
- Sort a list in ascending and descending order.
- Find the second largest and second smallest number in a list.
- Find common elements between two lists.
- Find the sum of all list elements.
- Create a list of squares using list comprehension.
- Separate even and odd numbers from a list.

B. Tuple Programs

- Create a tuple and access elements using indexing and slicing.
- Find the length of a tuple without using `len()`.
- Convert a list into a tuple and vice versa.
- Find the repeated elements in a tuple.
- Check if an element exists in a tuple.
- Find the maximum and minimum values in a tuple.

C. Set Programs

- Create a set and perform `add()`, `remove()`, `pop()` operations.
- Find union, intersection, and difference of two sets.
- Check subset, superset, and disjoint sets.
- Remove duplicates from a list using a set.
- Find common and unique elements between two sets.
- Create a frozen set and show its immutability.
- Find symmetric difference between sets.
- Check if two sets are equal.

D. Dictionary Programs

- Create a dictionary and perform insert, update, and delete operations.
 - Count the frequency of words in a string using a dictionary.
 - Sort a dictionary by keys and values.
 - Merge two dictionaries into one.
 - Create a dictionary from two lists (key–value pairs).
 - Find the key with the maximum value in a dictionary.
 - Check if a key exists in a dictionary.
 - Remove duplicate values in a dictionary.
 - Invert a dictionary (`value → key`).
 - Create a nested dictionary for student details.
 - Iterate through a dictionary using `items()`.
 - Convert a dictionary into a list of tuples and vice versa.
-

UNIT 3: Object-Oriented Python Programming

A. Class & Object Basics

- Create a simple class and object.
- Define a class with attributes and display them using an object.
- Demonstrate instance variables vs class variables.
- Create methods inside a class and call them through objects.
- Demonstrate the use of constructors (`__init__`).
- Create multiple objects for a class.
- Update and delete object properties.
- Count number of objects created for a class (using a class variable).

B. Encapsulation

- Demonstrate public, protected, and private attributes.
- Use getter and setter methods.
- Show name mangling with private variables.

C. Inheritance

- Demonstrate single inheritance.
- Demonstrate multilevel inheritance.
- Demonstrate multiple inheritance.
- Use `super()` in inheritance.
- Override parent class methods in child class.
- Show constructor overriding and constructor chaining.

D. Polymorphism

- Demonstrate method overloading (using default arguments).
- Demonstrate method overriding.
- Demonstrate operator overloading (`__add__`, `__lt__`, etc.).
- Overload comparison operators for a custom class.
- Use polymorphism with common interface (same method name in different classes).

E. Abstraction

- Create an abstract class using ABC module.
- Demonstrate abstract methods and subclass implementation.

F. Special Methods & Advanced OOP

- Implement `__str__()` and `__repr__()` methods.
- Overload indexing and slicing in classes (`__getitem__`).
- Create a class that behaves like an iterable (`__iter__`, `__next__`).

G. Exception Handling with OOP

- Handle exceptions inside class methods.
 - Create user-defined exceptions (custom exception class).
 - Demonstrate the use of try–except–finally inside OOP-based code.
-

UNIT 4: File Handling & Exception Handling

A. File Handling

- Create and write to a text file.
- Read a file and display contents.
- Append data to an existing file.
- Copy content from one file to another.
- Count lines, words, and characters in a file.
- Search for a word in a file.

B. Exception Handling

- Handle division by zero exception.
 - Handle multiple exceptions (ValueError, ZeroDivisionError, etc.).
 - Raise and handle user-defined exceptions.
 - File operation with exception handling.
-

UNIT 5: NumPy, Pandas & Matplotlib

A. NumPy

- Create and display a NumPy array.
- Perform arithmetic operations on NumPy arrays.
- Matrix operations using NumPy (transpose, inverse, determinant, dot product).
- Indexing, slicing, and reshaping of arrays.

B. Pandas

- Create and display Series and DataFrame.
- Perform data manipulation in DataFrame.
- Handle missing values in Pandas.

C. Matplotlib

- Plot basic charts (line, bar, scatter).
 - Plot histogram and pie chart.
 - Multiple subplots and styling.
-

UNIT 6: Searching & Sorting Algorithms

A. Searching

- Implement Linear Search in a list.
- Implement Binary Search in a sorted list.

B. Sorting

- Implement Bubble Sort.
 - Implement Selection Sort.
 - Implement Insertion Sort.
 - Implement Merge Sort.
-

UNIT 7: Case Studies(OOPS)

- **Student Management System** — Add student details, compute total & grade, display topper, save records in file.
 - **Library Management System** — Manage books, issue/return, check availability, fine calculation.
 - **Online Shopping Cart System** — Manage products, cart, total bill calculation, discounts, store purchase history.
 - **Banking System** — Manage accounts (savings/current), deposit, withdraw, minimum balance check, interest calculation.
 - **Employee Payroll System** — Manage employees, salary calculation, generate payslip, display highest salary.
-

UNIT 8: Special Functions in Python

- Demonstrate `len()` to find length of string, list, or tuple.
- Demonstrate `sum()` to calculate sum of elements in a list or tuple.
- Find maximum and minimum using `max()` and `min()`.
- Use `abs()` to find absolute value of a number.
- Use `round()` to round a floating-point number.
- Use `pow()` to calculate exponentiation.
- Demonstrate `sorted()` function on lists/tuples.
- Use `enumerate()` to iterate with index.
- Demonstrate `zip()` to combine multiple iterables.
- Use `map()` with a function to apply operation on list elements.
- Use `filter()` to filter elements based on a condition.
- Use `reduce()` from `functools` to perform cumulative operations.
- Demonstrate `lambda` (anonymous functions) with `map()` or `filter()`.
- Demonstrate `any()` and `all()` functions on a list of boolean values.
- Demonstrate `isinstance()` to check data type.