GLS UNIVERSITY

FACULTY OF COMPUTER APPLICATIONS & INFORMATION TECHNOLOGY

221601505 PRACTICALS ON PYTHON iMSCIT Sem – V

Unit 4 Practical Assignment

1. Create a `Hospital` system with `Doctor`, `Patient`, and `Appointment` classes. Implement methods to schedule appointments and print a daily appointment list.

2. Create a `TravelAgency` class that manages multiple `Trip` objects. Each trip contains destination, cost, and duration. Accept trips from the user and display the most expensive trip.

3. Build a `Restaurant` ordering system. Create classes for `MenuItem` and `Order`. Allow adding menu items and computing the total bill with tax. Access attributes via methods only (no direct attribute printing).

4. Create a `DatabaseConnection` class where the constructor simulates connecting to a database and the destructor automatically closes the connection and logs the closing time in a file.

5. Create a `Media` base class and a `Movie` subclass with additional attributes such as genre and rating. Add a method to recommend a movie based on rating and genre.

6. Implement an e-commerce hierarchy: `User` → `Customer` → `PremiumCustomer`. Add discounts and loyalty points calculations at each level.

7. Design a `SmartDevice` class that inherits from both `Phone` and `Camera`. Handle method name clashes using method resolution order.

8. Create a `Vehicle` base class with subclasses `Car`, `Bike`, and `Truck`. Each should override a method `calculate_trip_cost(distance)` using their own fuel efficiency.

9. Simulate an online ticket booking system where user inputs seat numbers. Handle errors for invalid seat number range, already booked seats, and invalid data types.

10. Create a program to read product details from a CSV file. Handle file not found, incorrect CSV format, and empty file errors with different exception blocks.

11. Create a `StudentResult` class that calculates grade percentages. Handle exceptions if marks entered exceed subject maximums or are negative.

12. Implement a `PasswordManager` class that raises custom exceptions (`WeakPasswordError`, `PasswordMismatchError`) when user creates or updates a password not following given security rules.

13. Write a `FileMerger` utility that merges multiple files into one. Use `try… finally` to ensure that all opened files close correctly even if an error occurs mid-process.

14. Build a `BankAccount` class with `withdraw` and `deposit` methods. Use `assert` to ensure withdrawal amount is positive, `raise` to throw `InsufficientBalanceError`, and `finally` to log every transaction attempt to a text file.

15. Create a program that uses the `with` statement to encrypt and decrypt a text file using a simple cipher. Handle errors if the file is missing or unreadable.

16. Develop a Ride Sharing Application with classes `Driver`, `Passenger`, `Ride`, and `Payment`. Include dynamic ride fare calculation based on distance and surge pricing, custom exceptions for invalid payment method or ride cancellation, and file handling to log completed rides using `with`.

17. Design a mini Library Management System with `Book`, `Member`, and `Transaction` classes to demonstrate attributes, methods, encapsulation, and object relationships. Add books, issue/return books, and track availability.