INSTITUTE OF TECHNOLOGY AND MANAGEMENT
SKILLS UNIVERSITY,
KHARGHAR, NAVI MUMBAI

PYTHON PROGRAMMING LAB

Prepared by:

Name of Student: __Shikha singh_____

Roll No: _25_____

Batch: 2023-27

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Exp. No	List of Experiment
	1.1 Write a program to compute Simple Interest.
	1.2 Write a program to perform arithmetic, Relational operators.
	1.3 Write a program to find whether a given no is even & odd.
	1.4 Write a program to print first n natural number & their sum.
	1.5 Write a program to determine whether the character entered is a Vowel or not .
	1.6 Write a program to find whether given number is an Armstrong Number.
	1.7 Write a program using for loop to calculate factorial of a No.
	1.8 Write a program to print the following pattern
	i) * * * * * * * * * * * * * * *
	ii) 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
	iii) *

	2.1 Write a program that define the list of defines the list of define countries that are in BRICS.
	2.2 Write a program to traverse a list in reverse order. 1.By using Reverse method. 2.By using slicing
	2.3 Write a program that scans the email address and forms a tuple of username and domain.
	2.4 Write a program to create a list of tuples from given list having number and add its cube in tuple. i/p: c= [2,3,4,5,6,7,8,9]
	2.5 Write a program to compare two dictionaries in Python? (By using == operator)
	2.6 Write a program that creates dictionary of cube of odd numbers in the range.
	2.7 Write a program for various list slicing operation. a= [10,20,30,40,50,60,70,80,90,100] i. Print Complete list ii. Print 4th element of list iii. Print list from0th to 4th index. iv. Print list -7th to 3rd element v. Appending an element to list. vi. Sorting the element of list. vii. Popping an element. viii. Removing Specified element. ix. Entering an element at specified index. x. Counting the occurrence of a specified element. xi. Extending list. xii. Reversing the list.
	3.1 Write a program to extend a list in python by using given approach. i. By using + operator. ii. By using Append ()

	iii. By using extend ()
	3.2 Write a program to add two matrices.
	3.3 Write a Python function that takes a list and returns a new list with distinct elements from the first list.
	3.4 Write a program to Check whether a number is perfect or not.
	3.5 Write a Python function that accepts a string and counts the number of upper- and lower-case letters. string_test= 'Today is My Best Day'
	4.1 Write a program to Create Employee Class & add methods to get employee details & print.
	4.2 Write a program to take input as name, email & age from user using combination of keywords argument and positional arguments (*args and**kwargs) using function,
	4.3 Write a program to admit the students in the different Departments(pgdm/btech) and count the students. (Class, Object and Constructor).
	4.4 Write a program that has a class store which keeps the record of code and price of product display the menu of all product and prompt to enter the quantity of each item required and finally generate the bill and display the total amount.
	4.5 Write a program to take input from user for addition of two numbers using (single inheritance).
	4.6 Write a program to create two base classes LU and ITM and one derived class. (Multiple inheritance).
	4.7 Write a program to implement Multilevel inheritance, Grandfather□Father-□Child to show property inheritance from grandfather to child.
	4.8 Write a program Design the Library catalogue system using inheritance take base class (library item) and derived class (Book, DVD & Journal) Each derived

	class should have unique attribute and methods and system should support Check in and check out the system. (Using Inheritance and Method overriding)
	5.1 Write a program to create my_module for addition of two numbers and import it in main script.
	5.2 Write a program to create the Bank Module to perform the operations such as Check the Balance, withdraw and deposit the money in bank account and import the module in main file.
	5.3 Write a program to create a package with name cars and add different modules (such as BMW, AUDI, NISSAN) having classes and functionality and import them in main file cars.
	6.1 Write a program to implement Multithreading. Printing “Hello” with one thread & printing “Hi” with another thread.
	7.1 Write a program to use ‘whether API’ and print temperature of any city, also print the sunrise and sunset times for the same humidity of that area.
	7.2 Write a program to use the ‘API’ of crypto currency.

Name of Student: Shikha singh_____

Roll Number: _25_____

Experiment No: 4.6

Title:

4.6 Write a program to create two base classes LU and ITM and one derived class.

(Multiple inheritance).

Theory:

Multiple inheritance in Python occurs when a class inherits from more than one base class. This allows a derived class to inherit attributes and methods from multiple parent classes. In Python, a class can inherit from multiple classes by listing them in the parentheses after the class name when defining the class.
In the given example:

- **The LU class has its data and a method to display it (display_lu).**
- **The ITM class has its data and a method to display it (display_itm).**
- **The Derived class inherits from both LU and ITM classes and has its own data and a method to display it (display_derived).**

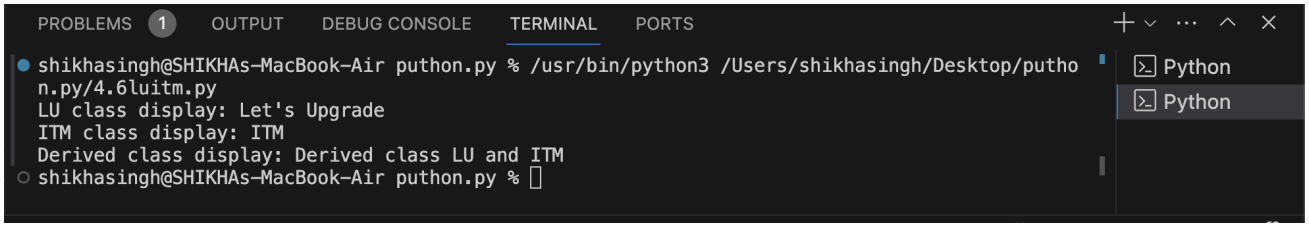
When an object of the Derived class is created, it can access methods and attributes from both LU and ITM classes, along with its own.

Code:

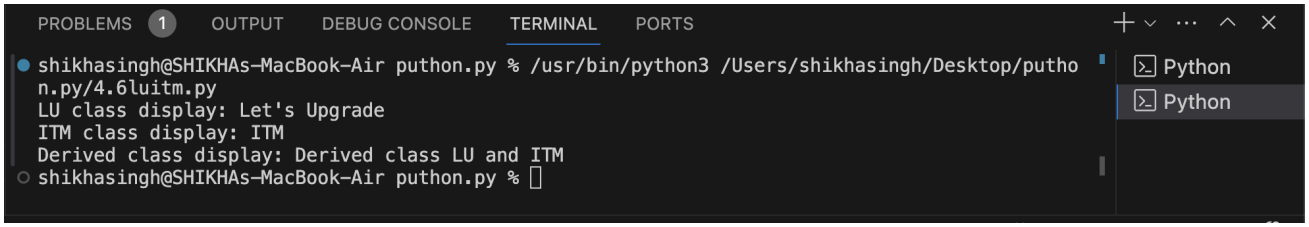
```
class LU:
    def __init__(self):
        self.lu_data = "Let's Upgrade"
    def display_lu(self):
        print("LU class display:", self.lu_data)
class ITM:
    def __init__(self):
        self.itm_data = "ITM "
    def display_itm(self):
        print("ITM class display:", self.itm_data)
class Derived(LU, ITM):
    def __init__(self):

        LU.__init__(self)
        ITM.__init__(self)
        self.derived_data = "Derived class LU and ITM"
    def display_derived(self):
        print("Derived class display:", self.derived_data)
derived_obj = Derived()
derived_obj.display_lu()
derived_obj.display_itm()
derived_obj.display_derived()
```

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Multiple inheritance can be a powerful feature in object-oriented programming, allowing a class to inherit from multiple sources. However, it should be used with caution to avoid potential issues like the diamond problem, where ambiguity arises due to the same method being present in multiple parent classes.
In Python, the method resolution order (MRO) is used to determine the order in which base classes are searched when a method is called on an object. This order is essential in cases of multiple inheritance.