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	1.1 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	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	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	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	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	6.1 Write a program to implement Multithreading. Printing “Hello” with one thread & printing “Hi” with another thread.
7.	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:

6.1

Title:

6.1 Write a program to implement Multithreading. Printing “Hello” with one thread & printing “Hi” with another thread

Theory:

Multithreading:

Multithreading is a programming concept where multiple threads (smaller units of a process) run concurrently within the same program. Each thread represents an independent flow of execution, allowing for better utilization of system resources and improved performance.

Python Threading Module:

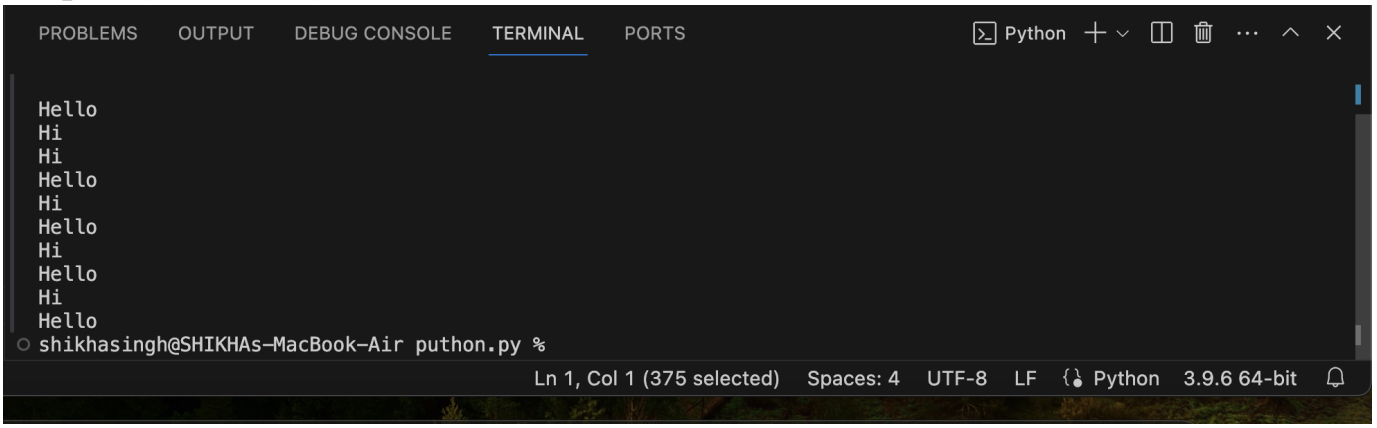
In Python, the threading module is used for multithreading. However, due to the Global Interpreter Lock (GIL) in CPython, threads are not suitable for parallel execution of CPU-bound tasks. The GIL restricts only one thread to execute Python bytecode at a time, preventing true parallelism. Despite this limitation, threading can be beneficial for I/O-bound tasks, such as network or file operations

Code:

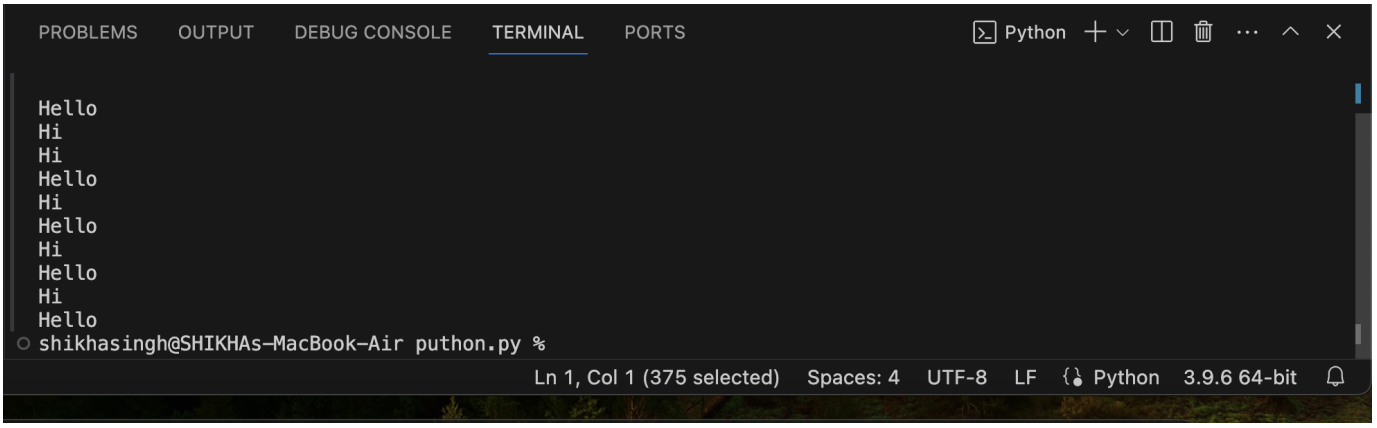
```
import threading
import time
def print_hello():
    for _ in range(5):
        print("Hello")
        time.sleep(0.5)
def print_hi():
    for _ in range(5):
        print("Hi")
        time.sleep(0.5)
thread_hello = threading.Thread(target=print_hello)
thread_hi = threading.Thread(target=print_hi)
thread_hello.start()
thread_hi.start()
thread_hello.join()
```

thread_hi.join()

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

GIL Limitation:

- **Python's GIL limits the effectiveness of multithreading in CPU-bound tasks. If your application involves heavy computational work, consider using the multiprocessing module for parallelism, as it bypasses the GIL by utilizing separate processes.**
- **Use in I/O-bound Tasks:**
- **Threading is more suitable for I/O-bound tasks where threads can perform non-blocking operations, such as waiting for data from a network or reading from a file. In such cases, threading can lead to performance improvements by allowing other threads to execute during waiting periods.**
- **Simple Example:**
- **The provided Python program demonstrates a basic use of threading. Two threads are created to print "Hello" and "Hi" alternatively. The time.sleep() function is used to simulate some processing time within each thread.**
- **Thread Safety:**
- **When multiple threads access shared resources, it's crucial to ensure thread safety. Synchronization mechanisms, like locks (threading.Lock), should be employed to prevent race conditions and data corruption.**
- **Considerations:**
- **Before implementing multithreading, carefully consider the nature of your tasks. If your application is highly parallelizable and involves CPU-intensive work, multiprocessing or alternative concurrency models may be more suitable.**