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

PYTHON PROGRAMMING LAB

Prepared by:

Name of Student: **Atharva Santosh Jadhav**

Roll No: **40**

Batch: **2023-27**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Exp. No	List of Experiment
1	1. Write a program to compute Simple Interest.
	2. Write a program to perform arithmetic, Relational operators.
	3. Write a program to find whether a given no is even & odd.
	4. Write a program to print first n natural number & their sum.
	5. Write a program to determine whether the character entered is a Vowel or not .
	6. Write a program to find whether given number is an Armstrong Number.
	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.

Experiment No: 1.1

Title: Write a program to compute Simple Interest.

Theory: Formula for Simple Interest=

Principal*rate of interest*time/100

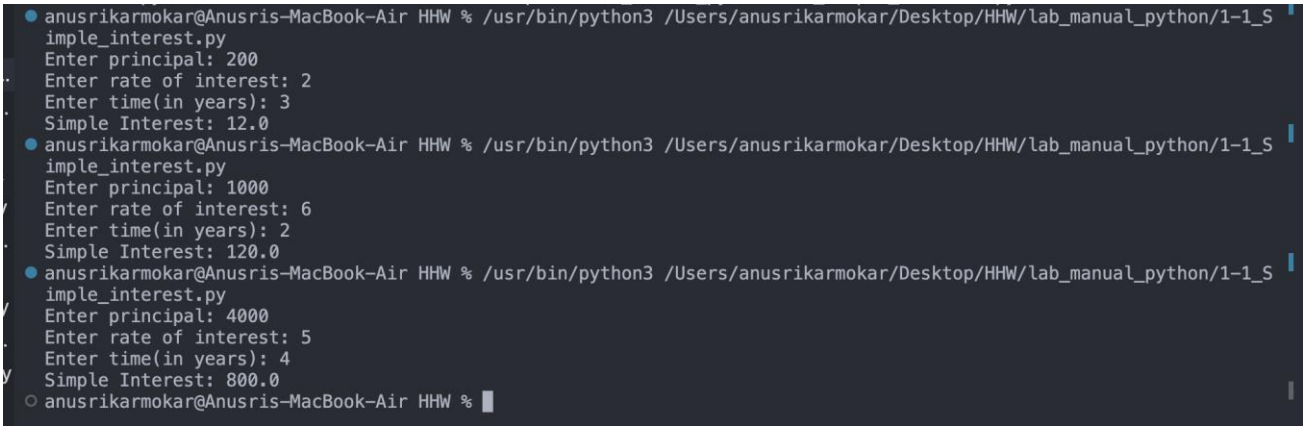
Since principal, rate of interest, and time can be in decimal points, their datatype is float. Taking all these values from the user, and then printing the value of simple interest from these values.

Code:

```
a=float(input("Enter principal: "))
b=float(input("Enter rate of interest: "))
c=float(input("Enter time(in years): "))
print("Simple Interest:",(a*b*c)/100)
```

Output: (screenshot)

+



Test Case: Any two (screenshot)

Conclusion:

Hence using the formula for simple interest, I have calculated it from the values given by the user and printed the result.

Experiment No: 1.2

Title: Write a program to perform Arithmetic, Relational operators.

Theory: Arithmetic operators are those operators which perform mathematical arithmetic operations on the values or variables(eg- addition(+), subtraction(-), division(/), etc.) Relational operators are used to compare two or more values or variables(eg- less than(<), greater than(>), equal to(==) or not equal to(!=), etc.)

Code:

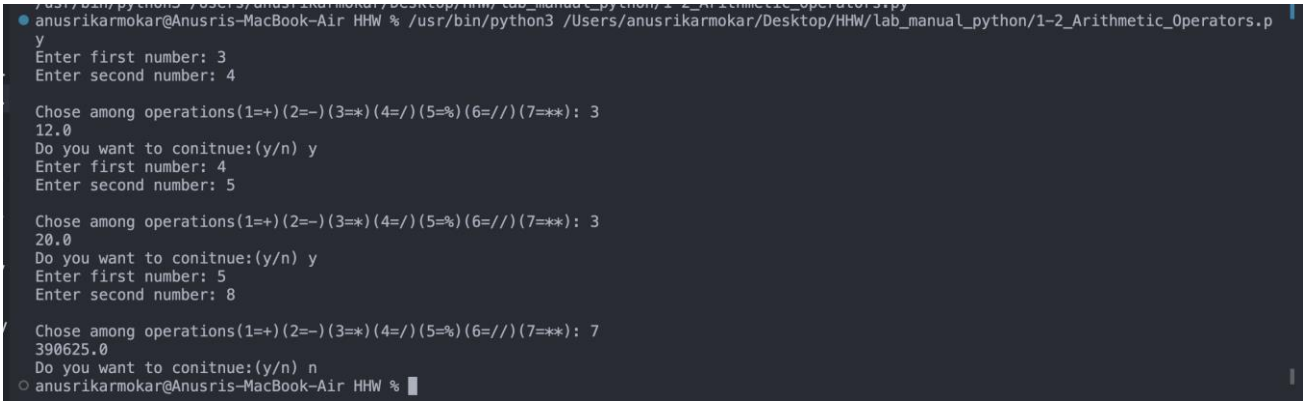
```
def calculator():
    checker="Y"
    while(checker=="Y" or checker=="y"):
        firstinput= float(input("Enter first number: "))
        secondinput= float(input("Enter second number: "))
        print()
        operation=int(input("Chose among operations(1=+)(2=-)(3=*)(4=/)(5=%)(6=//)(7=**): "))
        if(operation==1):
            print(firstinput+secondinput)
        elif(operation==2):
            if(firstinput>secondinput):
                print(firstinput-secondinput)
            else:
                print(secondinput-firstinput)
        elif(operation==3):
            print(firstinput*secondinput)
        elif(operation==4):
            if(firstinput>secondinput):
                print(firstinput/secondinput)
            else:
                print(secondinput/firstinput)
        elif(operation==5):
            if(firstinput>secondinput):
                print(firstinput%secondinput)
            else:
                print(secondinput%firstinput)
          #print(firstinput%secondinput)
        elif(operation==6):
            if(firstinput>secondinput):
                print(firstinput//secondinput)
            else:
```

```
            print(secondinput//firstinput)
        #print(firstinput//secondinput)
    elif(operation==7):
        print(firstinput**secondinput)
    else:
        if firstinput>secondinput:
            print(firstinput,"is greater than",secondinput)
        elif firstinput==secondinput:
            print(firstinput,"is equal to",secondinput)
        else:
            print(secondinput,"is greater than",firstinput)
    checker=str(input("Do you want to conitnue:(y/n) "))

calculator()
```

Output: (screenshot)

+

Test Case: Any two (screenshot)



Conclusion: Hence using the arithmetic, calculated the result of two values given by the user, andusing relational operators, printing the greater value given by the user.

Experiment No: 1.3

Title: Write a program to find whether a given no is even & odd.

Theory: Even numbers are those numbers which are divisible by 2(eg- 2,4,6,8,10, so on) and odd numbers are those numbers which are not divisible by 2(eg- 1,3,5,7,9, so on).

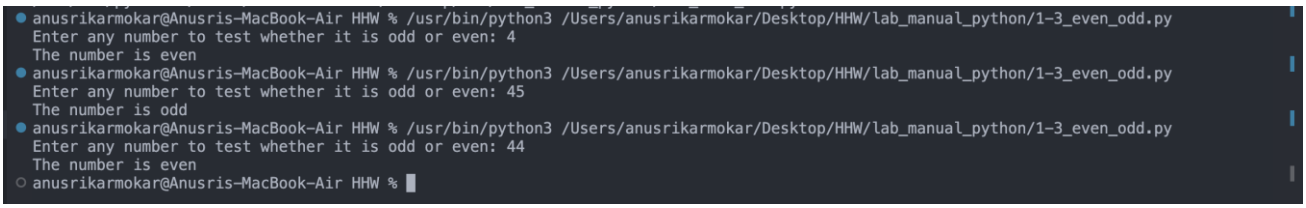
To check whether a given number is odd or even, we need to check whether it is divisible by 2 using modulus operator(%).

Code:

```
#even_odd
num = int (input ("Enter any number to test whether it is odd or even: "))
if (num % 2) == 0:
   print ("The number is even")
else:
   print ("The number is odd")
```

Output: (screenshot)

+



Test Case: Any two (screenshot)

Conclusion: Hence using the modulus operator(%), checked whether a given number is odd or even.

Experiment No: 1.4

Title: Write a program to print first n natural number & their sum.

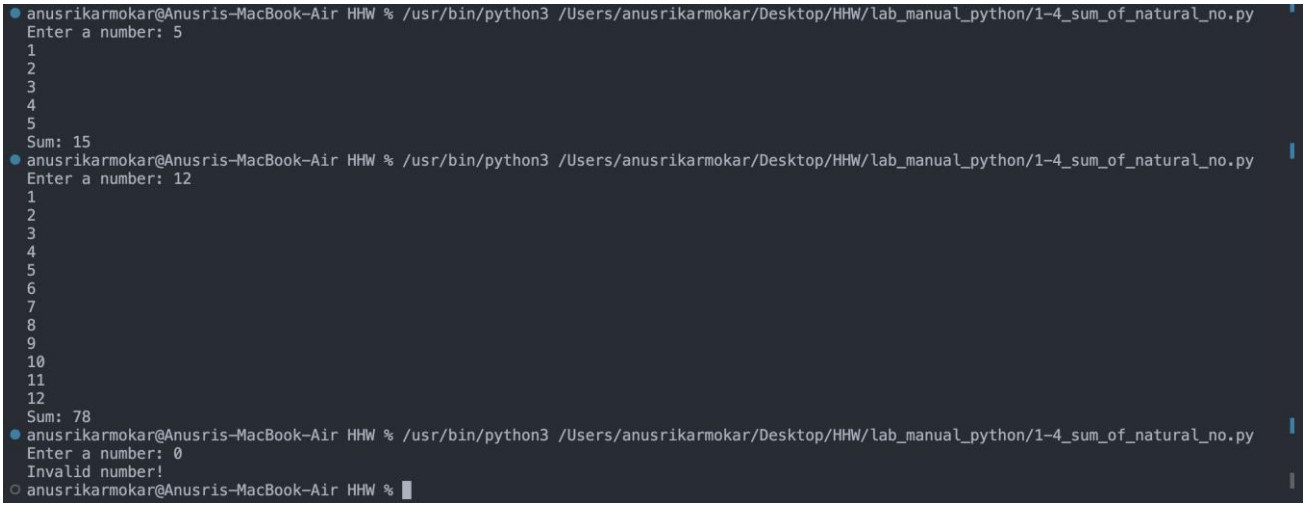
Theory: Natural numbers are numbers which start from 1 and go upto infinity. Using a for loop, print all the natural numbers and add them till the range given by the user.

Code:

```
a=int(input("Enter a number: "))
sum=0
if a<=0:
    print("Invalid number!")
else:
    for i in range(1,a+1):
        print(i)
        sum+=i
    print("Sum:",sum)
```

Output: (screenshot)

+



Test Case: Any two (screenshot)

Conclusion:

Hence using a for loop to iterate over each number from 1 to the range(given by the user which is inclusive), printed all the natural numbers and calculated their sum and printed it.

Experiment No: 1.5

Title: Write a program to determine whether the character entered is a Vowel or not.

Theory: A character is a vowel if it is either A,E,I,O,U(be it lower or upper case). Else, it is not a vowel(consonant).

Code:

```
a=["a","e","i","o","u"]
b=input("Enter a character: ")
if b.lower() in a:
    print(b,"is a vowel")
else:
    print(b,"is not a vowel")
```

Output: (screenshot)

+

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/1-5_check_vowel.py
Enter a character: A
A is a vowel
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/1-5_check_vowel.py
Enter a character: a
a is a vowel
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/1-5_check_vowel.py
Enter a character: z
z is not a vowel
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Test Case: Any two (screenshot)

Conclusion:

Hence using a membership operator(in) to check whether a user given character is inside the vowel list or not, and printing the appropriate message to the user.

Experiment No: 1.6

Title: Write a program to find whether given number is an Armstrong Number.

Theory: A positive number is called an Armstrong number of order n if

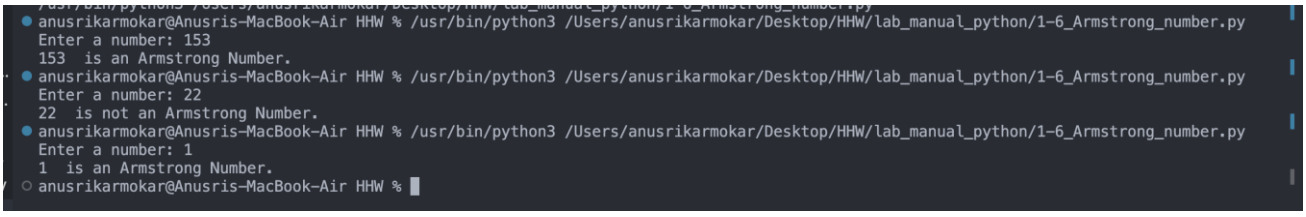
abcd=a^n+b^n+c^n+d^n. Eg- 153(3 digits are there, therefore, order=3), therefore, 1^3+5^3+3^3=153. Therefore, 153 is an Armstrong number of order 3.

Code:

```
num = int(input("Enter a number: "))
num_digits = len(str(num))

sum_of_digits = sum(int(digit) ** num_digits for digit in str(num))

if sum_of_digits == num:
   print(num," is an Armstrong Number.")
else:
   print(num," is not an Armstrong Number.")
```

Output: (screenshot)

+



Test Case: Any two (screenshot)

Conclusion:

Hence, calculating the order of the user given number(using len function), and adding the digits of the number raised to the power of the order(using while loop), and checking whether the sum is equal to the number(using relational operator ==) and printing the appropriate message(using if else statement).

Experiment No: 1.7

Title: Write a program using for loop to calculate factorial of a number.

Theory: Factorial of a number is the product of all numbers from the number till . It is denoted by n!, where n is the number given by the user. Only positive numbers have a

factorial.

Eg- 5!=5*4*3*2*1=120

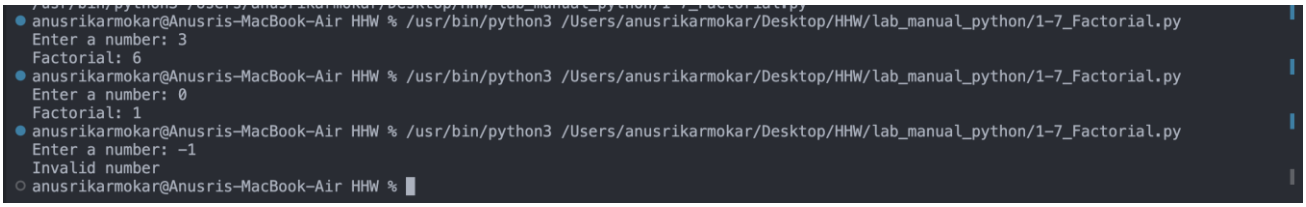
Code:

```
a=int(input("Enter a number: "))
fact=1
if a<0:
    print("Invalid number")
elif a==0 or a==1:
    print("Factorial:",1)
else:
    for i in range(1,a+1):
        fact*=i
```

```
    print("Factorial:",fact)
```

Output: (screenshot)

+



Test Case: Any two (screenshot)

Conclusion:

Hence, calculating the factorial of the user given number by first checking if it is a valid number(using if else statement), and using a for loop by calculating all the numbers from 1 to the number itself and printing it.

Experiment No: 1.8

Title: Write a program to print the following pattern

i)

*****

*** ***

*** * ***

*** * * ***

*** * * * ***

Theory:

Using nested for loops to print ascending half pyramid pattern. One for loop for the rows and another for loop to print stars in each row. Number of stars in each row should be equal to the row they are in. Eg- In 4th row, there should be 4 stars, in 5th row, there should be 5 stars, etc.

Code:

```
for i in range(1,6):
    for j in range(i):
        print(" * ",end='')
    print("\n")
```

Output: (screenshot)

Conclusion: Hence, printing a half pyramid ascending star pattern using nested for loops.

Experiment No: 1.8

Title: Write a program to print the following pattern

ii)

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

Theory: Using nested for loops to print ascending half pyramid pattern. One for loop for the rows and another for loop to print numbers in each row. There should be n numbers and all numbers should have n value in nth row. Eg- In 4th row there should be 4 numbers and each one of them should be of the value 4.

Code:

```
for i in range(1,6):
    for j in range(i):
        print(i,end='')
    print("\n")
```

Output: (screenshot)

Conclusion:

Hence, printing a half pyramid ascending number pattern using nested for loops.

Experiment No: 1.8

Title: Write a program to print the following pattern

iii)

*****

* * *

* * * * *

* * * * * * *

* * * * * * * * *

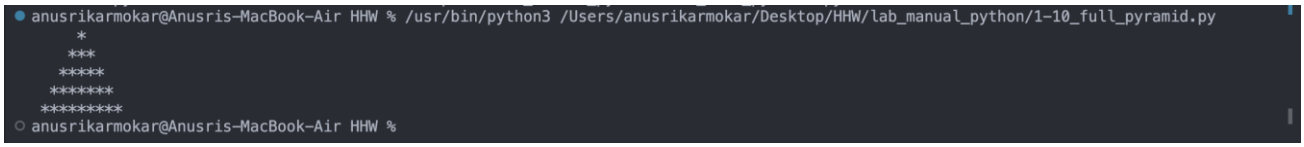
Theory: Using nested for loops to print ascending pyramid pattern. One for loop for the rows and another for loop to print stars in each row. There should be 2n-1 stars in nth row.

Eg- In 4th row there should be 2(4)-1=7 stars, in 5th row there should be 2(5)-1=9 stars.

Code:

```
for i in range(1,6):
    for j in range(6,i,-1): #pyramid pattern ascending
        print(" ",end='')
    for k in range(2*i-1):
        print("*",end='')
    print()
```

Output(Screenshot):



Conclusion: Hence, printing a half pyramid ascending number pattern using nested for loops.

Experiment No: 2.1

Title: Write a program that define the list of defines the list of define countries that are in BRICS.

Theory: BRICS countries= Brazil, Russia, India, China, South Africa.

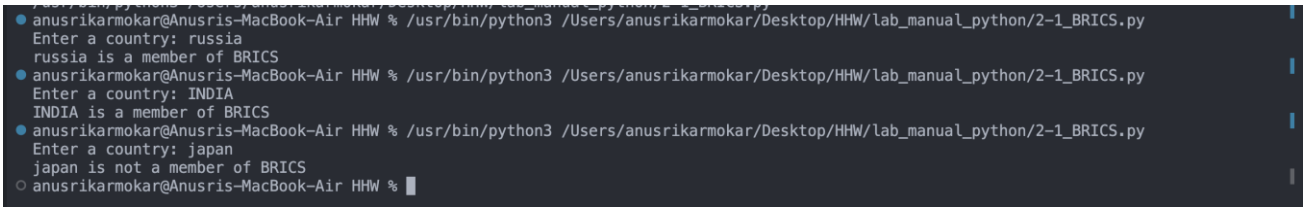
Using a membership operator(in), can check whether country given by user is a BRICS member or not. Can use in operator to check whether the value of variable is inside BRICS list or not.

Code:

```
brics=["brazil","russia","india","china","south africa"]
a=input("Enter a country: ")
if a.lower() in brics:
   print(a,"is a member of BRICS")
else:
   print(a,"is not a member of BRICS")
```

Output(Screenshot):

+



Test Case: Any two (screenshot)

Conclusion: Hence, using a membership operator(in), check user given country is in BRICS or not and printing the appropriate message using if else statement.

Experiment No: 2.2

Title: Write a program to traverse a list in reverse order.

1.By using Reverse method.

2.By using slicing

Theory: 1. By using predefined function reverse(), we can reverse a list and print it.

2. By using index slicing, we can print the list in reverse using step size as -1.

Code:

1.

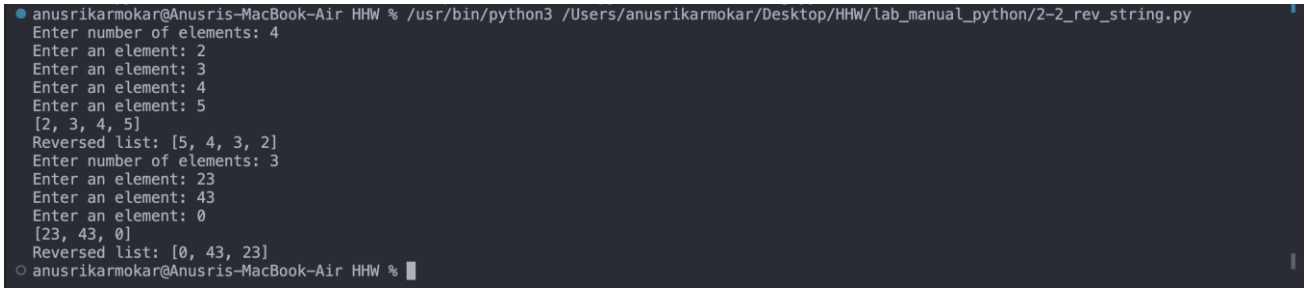
```
a=[]
n=int(input("Enter number of elements: "))
for i in range(n):
    b=int(input("Enter an element: "))
    a.append(b)
print(a)
a.reverse()
print("Reversed list:",a)
```

2.

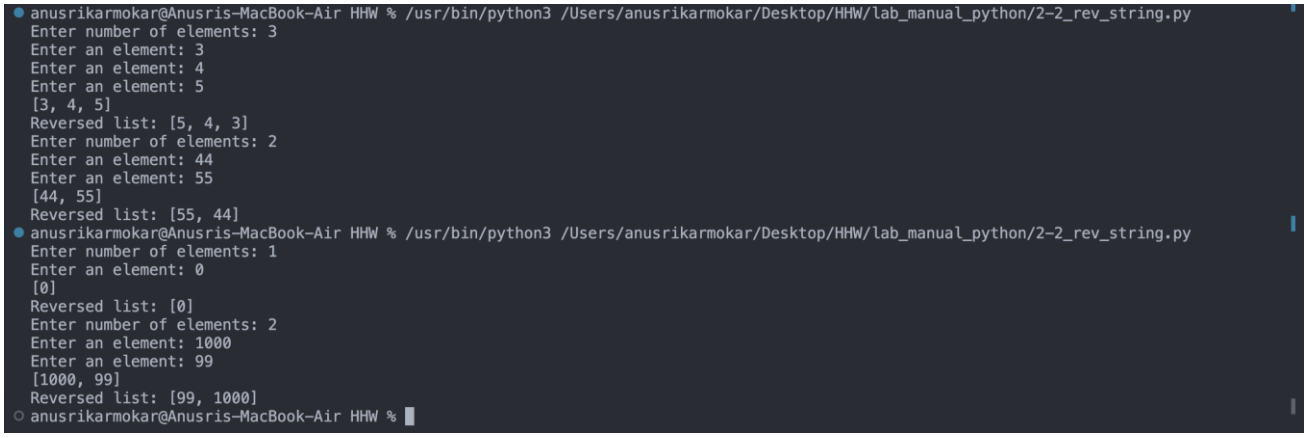
```
a=[]
n=int(input("Enter number of elements: "))
for i in range(n):
    b=int(input("Enter an element: "))
    a.append(b)
```

```
print(a)
print("Reversed list:",a[::-1])
```

Output(Screenshot):



Conclusion: Hence, traversing the index in reverse order using index slicing and predefined function reverse().



Experiment No: 2.3

Title: Write a program that scans the email address and forms a tuple of username and domain.

Theory:

Using split() function with @ as a parameter to split the email address given by user and store it in a tuple. Then print the first element of the tuple as username and the second element from the same tuple as domain.

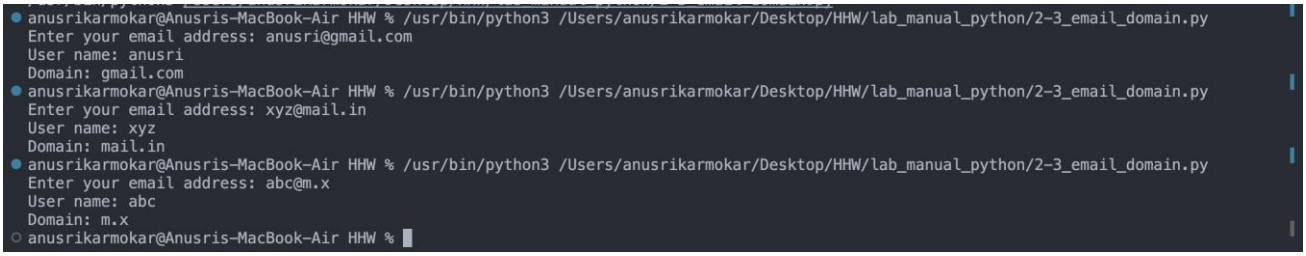
Code:

```
a=input("Enter your email address: ")
if "@" in a:
    b=(a.split("@"))
    c=(b[0],b[1])
    print("User name:",c[0])
    print("Domain:",c[1])
else:
    print("Wrong email address entered")
```

Output(Screenshot):

+

Test Case: Any two (screenshot)



Conclusion:

Hence, using split() function to split the email address in username and domain and printing them after storing them in a tuple.

Experiment No: 2.4

Title: Write a program to create a list of tuples from given list having number and add its cube in tuple.

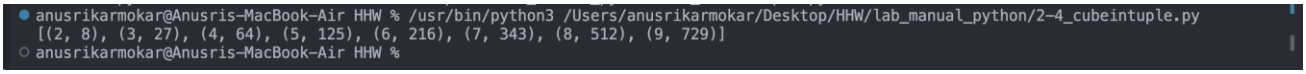
i/p: c= [2,3,4,5,6,7,8,9]

Theory: Using list comprehension, make another list with tuples as its elements. Each tuple would have the number from original list and it's cube using arithmetic operator(**).

Code:

```
c= [2,3,4,5,6,7,8,9]
a=[(num,num**3)for num in c]
print(a)
```

Output(Screenshot):



Conclusion: Hence, using list comprehension to make a list of tuples containing the values and their cube using ** operator from the original list.

Experiment No: 2.5

Title: 2.5 Write a program to compare two dictionaries in Python

(By using == operator)

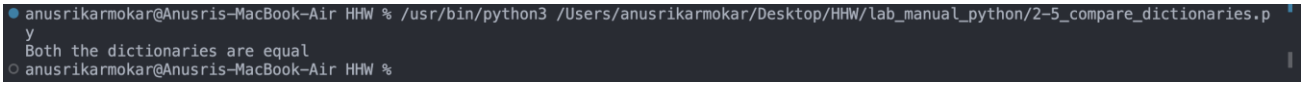
Theory: Dictionary is a datatype in Python which stores information as key-value pairs, where keys are unique and are used to distinguish between values. Using relational

operator(==) to check whether two dictionaries have same key-value pairs or not.

Code:

```
a={1:1,2:"a",3:23,4:52}
b={1:1,2:"a",3:23,4:52}
if a==b:
    print("Both the dictionaries are equal")
else:
    print("Both the dictionaries are not equal")
```

Output(Screenshot):



Conclusion: Hence, using == operator to check whether two dictionaries have same key-value pairsor not and printing appropriate messages using if else statements.

Experiment No: 2.6

Title: Write a program that creates dictionary of cube of odd numbers in the range.

Theory: Using for loop to iterate over every number from 1 till range given by user and using if statement to check whether a number is odd or even using modulus operator(%) and making the number as key and its cube as a value, and printing the dictionary in the end.

Code:

```
a=int(input("Enter range: "))
b={}
for i in range(1,a):
    if i%2!=0:
        b[i]=i**3
print(b)
```

Output(Screenshot):

+

Test Case: Any two (screenshot)

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/2-6_cube_of_oddnum.py
Enter range: 3
{1: 1}
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/2-6_cube_of_oddnum.py
Enter range: 5
{1: 1, 3: 27}
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/2-6_cube_of_oddnum.py
Enter range: 7
{1: 1, 3: 27, 5: 125}
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Conclusion:

Hence, using for loop and if statement to create a dictionary of odd numbers as keys and their cube as values and printing the dictionary.

Experiment No: 2.7

Title: 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.

Theory: Using index slicing to print specific elements, predefined functions such as append() to add an element in the list, sort() to sort the list in ascending or descending order, pop() to remove and element in the list, insert() to add an element in the list at a specific index, remove() to remove a specific element from the list, for loop to check the occurrence of an element, extend() to add multiple elements in the list, reverse() to reverse the list.

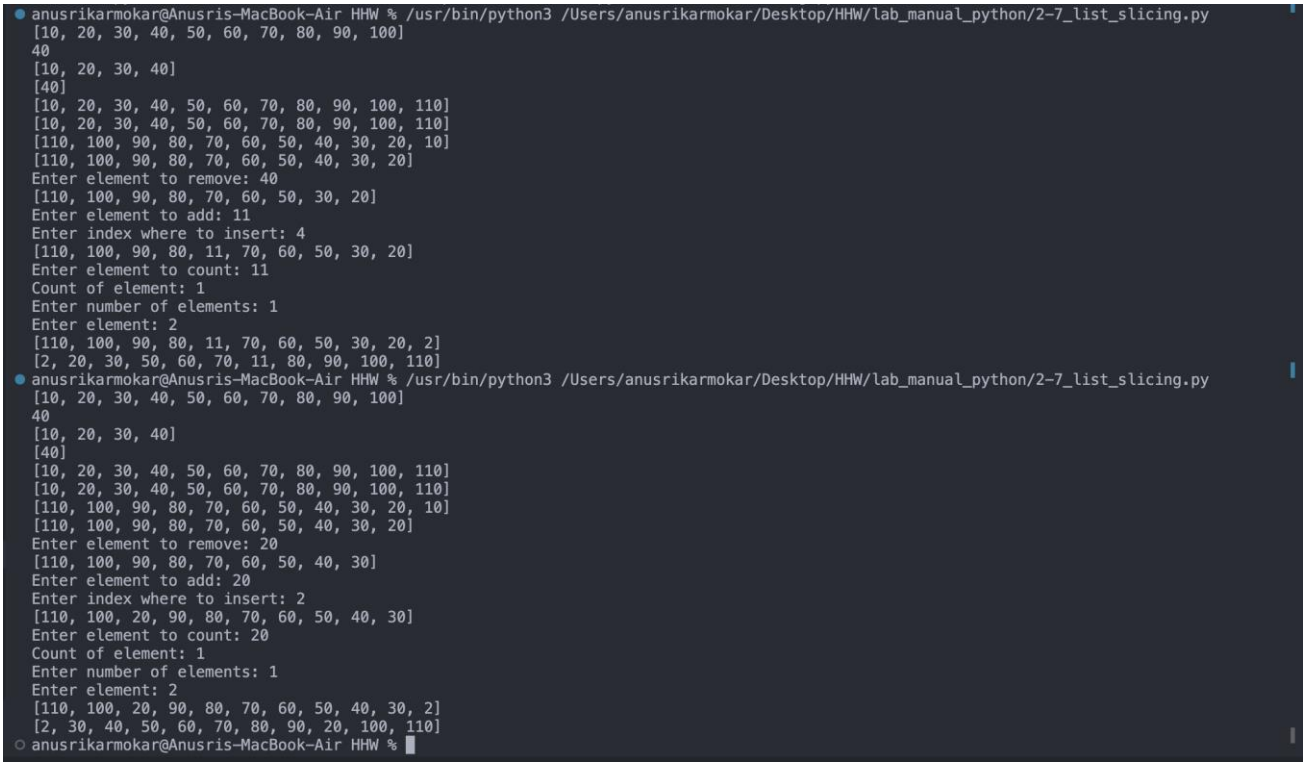
Code:

```
a= [10,20,30,40,50,60,70,80,90,100]
print(a)

print(a[3]) #Print 4th element of list

print(a[0:4]) #Print list from0th to 4th index.

print(a[-7:4]) #Print list -7th to 3rd element

a.append(110) #Appending an element to list.
print(a)

a.sort() #Sorting the element of list.
print(a)
a.sort(reverse=True) #Sorting the element of list in descending order.
print(a)

a.pop() #Popping an element.
```

```
print(a)

b=int(input("Enter element to remove: "))
if b in a:
    a.remove(b) #Removing Specified element.
    print(a)
else:
    print("Invalid element")

b=int(input("Enter element to add: "))
c=int(input("Enter index where to insert: "))
a.insert(c,b) #Entering an element at specified index.
print(a)

b=int(input("Enter element to count: "))
count=0
for i in a:
    if i==b:
        count+=1 #Counting the occurrence of a specified element.
print("Count of element:",count)

c=int(input("Enter number of elements: "))

d=[]
for i in range(c):
    b=int(input("Enter element: "))
    d.append(b)
a.extend(d) #Extending list.
print(a)

a.reverse() #Reversing the list.
print(a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/2-7_list_slicing.py
[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
40
[10, 20, 30, 40]
[40]
[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]
[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]
[110, 100, 90, 80, 70, 60, 50, 40, 30, 20, 10]
[110, 100, 90, 80, 70, 60, 50, 40, 30, 20]
Enter element to remove: 30
[110, 100, 90, 80, 70, 60, 50, 40, 20]
Enter element to add: 11
Enter index where to insert: 2
[110, 100, 11, 90, 80, 70, 60, 50, 40, 20]
Enter element to count: 20
Count of element: 1
Enter number of elements: 2
Enter element: 22
Enter element: 33
[110, 100, 11, 90, 80, 70, 60, 50, 40, 20, 22, 33]
[33, 22, 20, 40, 50, 60, 70, 80, 90, 11, 100, 110]
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):



Test Case: Any two (screenshot)

Conclusion:Hence, using for loop, slicing, and various predefined list functions to add, remove, sort and reverse a list.

Experiment No: 3.1

Title: 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 ()

Theory:

+ operator is used to add a list to another list. Append() is used to add an element at

the end of the list. Extend() is used to add multiple elements at the end of the list.

Code:

```
a=[1,2,3,4,5]
c=[]
b=int(input("Enter number of elements: "))
for i in range(b):
    d=int(input("Enter element: "))
    c.append(d)
a+=c #using + operator
print(a)

b=int(input("Enter element: "))
a.append(b) #using append()
print(a)

c=[]
b=int(input("Enter number of elements: "))
for i in range(b):
    d=int(input("Enter element: "))
    c.append(d)
a.extend(c) #using extend()
print(a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-1_extend_list.py
Enter number of elements: 2
Enter element: 33
Enter element: 44
[1, 2, 3, 4, 5, 33, 44]
```

Output(Screenshot):

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-1_extend_list.py
Enter number of elements: 2
Enter element: 22
Enter element: 33
[1, 2, 3, 4, 5, 22, 33]
```

Test Case: Any two (screenshot)

```
Enter element: 1
[1, 2, 3, 4, 5, 22, 33, 1]
```

Conclusion: Hence, using for loop and if statement to create a dictionary of odd numbers as keys and their cube as values and printing the dictionary.

Experiment No: 3.2

Title: Write a program to add two matrices.

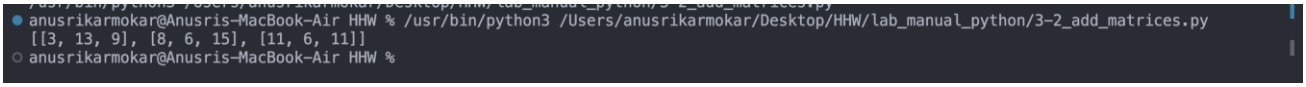
Theory: A matrix is a nested list in Python(a list of lists). It consists of two lists which have three lists inside each of them. Using nested for loop, one for each of the two lists, add elements from the two matrices and store the sum in another matrix of same dimension and print it.

Code:

```
a=[[2,5,4],[1,3,9],[7,6,2]]
b=[[1,8,5],[7,3,6],[4,0,9]]
c=[[0,0,0],[0,0,0],[0,0,0]]
for i in range(len(a)):
    for j in range(len(b)):
        c[i][j]=a[i][j]+b[i][j]
print(c)
```

Output(Screenshot):

+



Test Case: Any two (screenshot)

Conclusion: Hence, using nested for loop, add the elements of two nested lists and store the sum in another nested list

Experiment No: 3.3

Title: Write a Python function that takes a list and returns a new list with distinct elements from the first list.

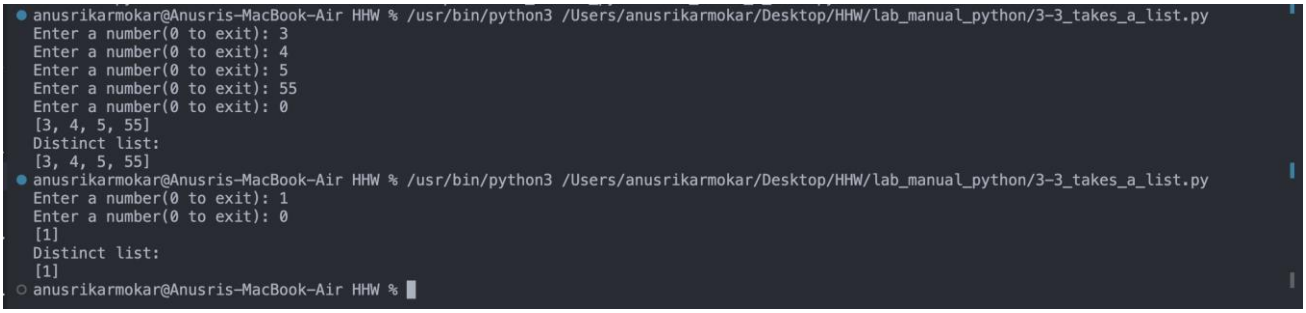
Theory: Making a user defined function distin() taking a user list as a parameter and then making an empty list and using a for loop to iterate through the user list and using membership operator(in) to check whether the element is present in the empty list too. If it is not present, append it in the list and check the next element for the same.

Code:

```
def distin(a):
    b=[]
    for i in a:
        if i not in b:
            b.append(i)
    print(b)
a=[]
while True:
    c=int(input("Enter a number(0 to exit): "))
    if c == 0:
        break
    else:
        a.append(c)
print(a)
print("Distinct list:")
distin(a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-3_takes_a_list.py
Enter a number(0 to exit): 3
Enter a number(0 to exit): 4
Enter a number(0 to exit): 5
Enter a number(0 to exit): 0
[3, 4, 5]
Distinct list:
[3, 4, 5]
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):



Test Case: Any two (screenshot)

Conclusion: Hence, by making a function, using nested for loop to check whether each element in the list is only present once and appending it to another list and then printing the other list at the end.

Experiment No: 3.4

Title: Write a program to Check whether a number is perfect or not.

Theory: A number is a perfect number if the sum of its divisors(excluding the number itself) is equal to the number itself. Eg- 6 is a perfect number as divisors of 6 are 1,2,3. Sum- 1+2+3=6 which is equal to the number itself.

Code:

```
def perfect(a):
    sum = 0
    for i in range(1, a):
        if a % i == 0:
            sum += i
    if sum == a:
        print(a, "is a perfect number")
    else:
        print(a, "is not a perfect number")
```

```
a = int(input("Enter a number: "))
perfect(a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-4_perfect_no.py
Enter a number: 33
33 is not a perfect number
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-4_perfect_no.py
```

Output(Screenshot):

Test Case: Any two (screenshot)

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-4_perfect_no.py
Enter a number: 28
28 is a perfect number
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/3-4_perfect_no.py
Enter a number: 99
99 is not a perfect number
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Conclusion: Hence, using a for loop to iterate over each number from 1 to number itself(exclusive) and checking whether it is a divisor of the number using if else statement and adding it in sum variable and checking afterwards if the sum is equal to the number and printing the appropriate message using if else statement.

Experiment No: 3.5

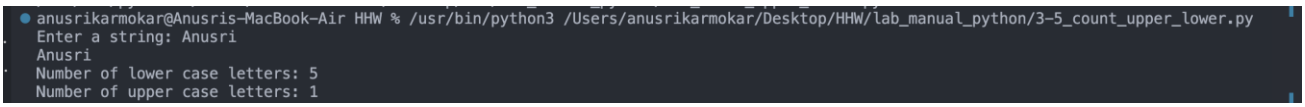
Title: Write a Python function that accepts a string and counts the number of upper-case and lower-case letters.

Theory: Using a for loop to iterate each character in the string and if elif statement and predefined functions isupper() and islower() to check whether a character is lowercase or uppercase and incrementing the value of the respective uppercase or lowercase counter by 1.

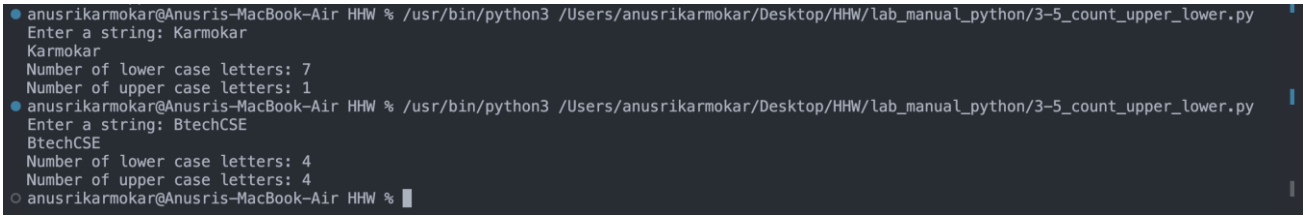
Code:

```
a=input("Enter a string: ")
ucount=0
lcount=0
for i in a:
   if i.islower():
      lcount+=1
   elif i.isupper():
      ucount+=1
print(a)
print("Number of lower case letters:",lcount)
print("Number of upper case letters:",ucount)
```

Output(Screenshot):



Test Case: Any two (screenshot)



Conclusion: Hence, using a for loop to iterate over each character in the string and using isupper() and islower() to check the characters and increment the counter variables value by 1 using if elif statement.

Experiment No: 4

Title: Write a program to Create Employee Class & add methods to get employee details & print.

Theory: A class is a blueprint for objects. It contains attributes and methods which the objects can access and use. An object is an instance of a class. Each class has its own copy of attributes and share the methods of the class.

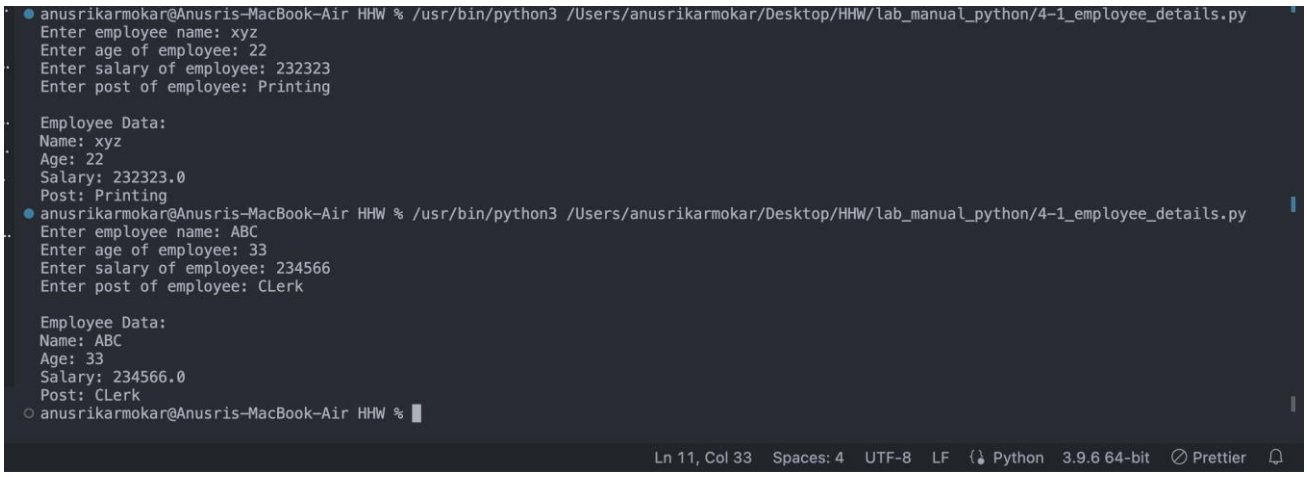
Code:

```
class Employee:
    __name=""
    __age=0
    __salary=0
    _post=""
    def setData(self):
        name=input("Enter employee name: ")
        self.__name=name
        age=int(input("Enter age of employee: "))
        self.__age=age
        salary=float(input("Enter salary of employee: "))
        self.__salary=salary
        post=input("Enter post of employee: ")
        self.__post=post
    def getData(self):
        print("\nEmployee Data:")
        print("Name:", self.__name)
        print("Age:", self.__age)
        print("Salary:", self.__salary)
        print("Post:", self.__post)
a=Employee()
a.setData()
a.getData()
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-1_employee_details.py
Enter employee name: Anusri
Enter age of employee: 18
Enter salary of employee: 2000000
Enter post of employee: Manager

Employee Data:
Name: Anusri
Age: 18
Salary: 2000000.0
Post: Manager
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):

Test Case: Any two (screenshot)



Conclusion: Hence, using get and set data functions to take values from the user and then assigning the attributes of the object these values and printing them to the user.

Experiment No: 4.2

Title: Write a program to take input as name, email & age from user using combination of keywords argument and positional arguments (*args andkwargs) using function.**

Theory: While defining a function, we use *args and **kwargs as parameters when we don't know how many arguments the user will pass during function call. *args will take a tuple of positional arguments as parameter and work on it. **kwargs will take a dictionary of keyword arguments as parameter and assign

the keys to keywords in arguments and the values as the values passed by the user in the arguments.

Code:

```
def kbfunction(*args, **kwargs):
    for i in args:
        print("Name:",i)
    for i in kwargs:
        print(i,":",kwargs[i])
b=input("Enter your name: ")
c=int(input("Enter your age: "))
a=input("Enter your email: ")
kbfunction(b,age=c,email=a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-2_keyword-positional_arg
s.py
Enter your name: Anusri
Enter your age: 18
Enter your email: anusri@gmail.com
Name: Anusri
age : 18
email : anusri@gmail.com
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-2_keyword-positional_arg
s.py
Enter your name: Piyush
Enter your age: 23
Enter your email: piyush@gmail.com
Name: Piyush
age : 23
email : piyush@gmail.com
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-2_keyword-positional_arg
s.py
Enter your name: Atharva
Enter your age: 22
Enter your email: atharva
Name: Atharva
age : 22
email : atharva
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Test Case: Any two (screenshot)

Conclusion: Hence, using dynamic argument passing(*args and **kwargs) to take multiple arguments from the user in a combination of positional and keyword arguments and printing them using a user defined function.

Experiment No: 4.3

Title: Write a program to admit the students in the different Departments(pgdm/btech) and count the students. (Class, Object and Constructor).

Theory: A constructor is automatically called when an object is created for a class. It is defined by __init__(self). It is used to initialise all the attributes of a class and self is a pointer which is used to tell the program to make changes only in a particular object and not access data or modify other objects.

Code:

```
class ITM:
    count=0
    bcount=0
    pcount=0
    def __init__(self):
        self.count=0
```

```
        self.bcount=0
        self.pcount=0
    def getData(self):
        a=input("Enter student name:" )
        b=int(input("Enter age: "))
        c=input("Enter address: ")
        d=int(input("Select department(1/2):\n1.BTECH\n2.PGDM\n"))
        while not(d==1 or d==2):
            print("Invalid choice\n")
            d=int(input("Select department(1/2):\n1.BTECH\n2.PGDM\n"))
        if d==1:
            self.dep="BTECH"
            ITM.bcount+=1
        elif d==2:
            self.dep="PGDM"
            ITM.pcount+=1
        self.name=a
        self.age=b
        self.address=c
        ITM.count+=1
    def setData(self):
        print("Student Details:\n")
        print("Name:",self.name)
        print("\nAge:",self.age)
        print("\nAddress:",self.address)
        print("\nDepartment:",self.dep)
a=int(input("Welcome to ITM. Press:\n1.Enter Student Data\n2.Number of admissions\n3.Display
students data\n4.Exit\n"))
objs=list()
while a!=4:
    if a==1:
        d=int(input("Enter number of students: "))
        for i in range(d):
            objs.append(ITM())
        for i in range(d):
            objs[i].getData()
        a=int(input("\nPress\n1.Enter another data\n2.Number of admissions\n3.Display student
details\n4.Exit\n"))
    elif a==2:
        b=int(input("Press\n1.BTECH admissions\n2.PGDM admissions\n"))
        while not (b==1 or b==2):
            print("Invalid choice\n")
            b=int(input("Press\n1.BTECH admissions\n2.PGDM admissions\n"))
        if b==1:
            print("Admissions of BTECH Done:",ITM.bcount)
            print("Total admissions:",ITM.count)
            a=int(input("\nPress\n1.Enter student data\n2.Number of admissions\n3.Display student
details\n4.Exit\n"))
        elif b==2:
            print("Admissions of PGDM Done:",ITM.pcount)
```

```
            print("Total admissions:",ITM.count)
            a=int(input("\nPress\n1.Enter student data\n2.Number of admissions\n3.Display student details\n4.Exit\n"))
        elif a==3:
            b=int(input("\nPress\n1.For BTECH students data\n2.For PGDM students data\n"))
            if b==1:
                for i in range(d):
                    if objs[i].dep=="BTECH":
                        objs[i].setData()
                a=int(input("\nPress\n1.Enter student data\n2.Number of admissions\n3.Display student details\n4.Exit\n"))
            elif b==2:
                for i in range(d):
                    if objs[i].dep=="PGDM":
                        objs[i].setData()
                a=int(input("\nPress\n1.Enter student data\n2.Number of admissions\n3.Display student details\n4.Exit\n"))
            else:
                print("Invalid choice")
                a=int(input("\nPress\n1.Enter student data\n2.Number of admissions\n3.Display student details\n4.Exit\n"))
        elif a==4:
            break
        else:
            print("Invalid choice")
            break
```

Output(Screenshot):

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-3_student_department.py
Welcome to ITM. Press:
1.Enter Student Data
2.Number of admissions
3.Display students data
4.Exit
1
Enter number of students: 2
Enter student name:Anusri
Enter age: 18
Enter address: sdvfr
Select department(1/2):
1.BTECH
2.PGDM
1
Enter student name:Sakshi
Enter age: 23
Enter address: dfvgb
Select department(1/2):
1.BTECH
2.PGDM
2

Press
1.Enter another data
2.Number of admissions
3.Display student details
4.Exit
2
Press
1.BTECH admissions
2.PGDM admissions
1
Admissions of BTECH Done: 1
Total admissions: 2

Press
1.Enter student data
2.Number of admissions
3.Display student details
4.Exit
3

Press
1.For BTECH students data
2.For PGDM students data
```

Ln 25, Col 23 Spaces: 4 UTF-8 LF Python 3.9.6 64-bit Prettier

Conclusion: Hence, using constructors and get and set functions and using while loop to print a menu to the user and call functions as per the user's need and making a list of objects to store information of multiple students of different branches.

```
Press
1.For BTECH students data
2.For PGDM students data
2
Student Details:

Name: Sakshi

Age: 23

Address: dfvgb

Department: PGDM

Press
1.Enter student data
2.Number of admissions
3.Display student details
4.Exit
```

Experiment No: 4.4

Title: 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.

Theory: Using constructor to print a menu of items to the user and getting quantity of the product and printing the bill with total amount at the end.

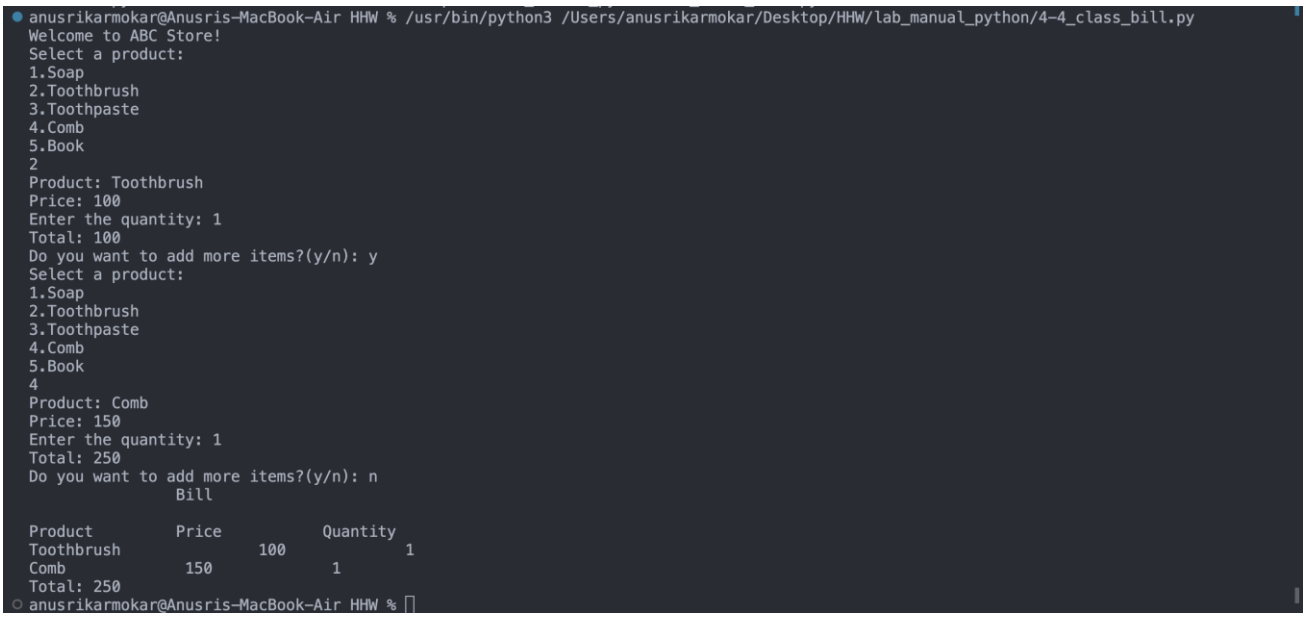
Code:

```
class Store:
    __itemCode=0
    __price=0
    __total=0
    product=[]
    quantity=[]
    price=[]
    def __init__(self):
        print("Welcome to ABC Store!")
        while True:
            a=int(input("Select a product:\n1.Soap\n2.Toothbrush\n3.Toothpaste\n4.Comb\n5.Book\n"))
            c=("Soap","Toothbrush","Toothpaste","Comb","Book")
            if a==1:
                self.__itemCode=1
                self.__price=50
            elif a==2:
                self.__itemCode=2
                self.__price=100
            elif a==3:
                self.__itemCode=3
                self.__price=200
            elif a==4:
                self.__itemCode=4
                self.__price=150
            elif a==5:
                self.__itemCode=5
                self.__price=500
            print("Product:",c[a-1],"\nPrice:",self.__price)
```

```
            self.product.append(c[a-1])
            self.price.append(self.__price)
            b=int(input("Enter the quantity: "))
            while b<=0:
                print("Invalid quantity")
                b=int(input("Enter the quantity: "))
            self.quantity.append(b)
            self.__total+=self.__price*b
            print("Total:",self.__total)
            d=input("Do you want to add more items?(y/n): ")
            if d.lower()=="n":
                break
        print("\t\tBill\n\nProduct\t\tPrice\t\tQuantity")
        for i in range(len(self.product)):
            print(self.product[i],"\t\t",self.price[i],"\t\t",self.quantity[i],"")
        print("Total:",self.__total)
obj=Store()
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-4_class_bill.py
Welcome to ABC Store!
Select a product:
1.Soap
2.Toothbrush
3.Toothpaste
4.Comb
5.Book
1
Product: Soap
Price: 50
Enter the quantity: 2
Total: 100
Do you want to add more items?(y/n): n
                Bill

Product         Price           Quantity
Soap             50              2
Total: 100
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):



Test Case: Any two (screenshot):

Conclusion: Hence, using constructor, printing a bill with total amount at the end.

Experiment No: 4.5

Title: Write a program to take input from user for addition of two numbers using (single inheritance).

Theory:

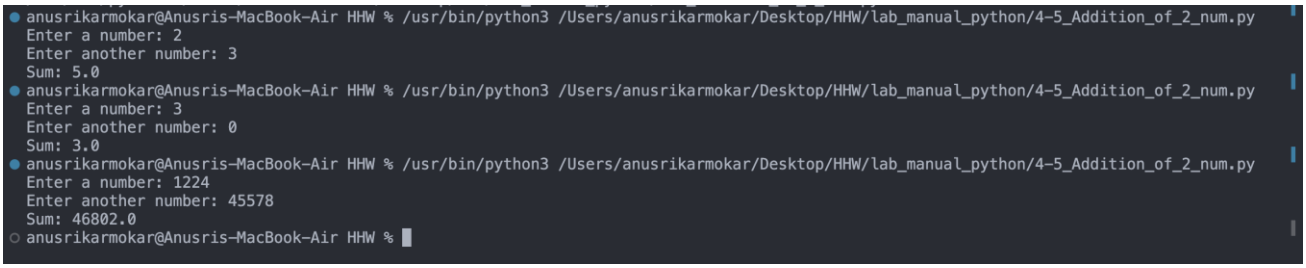
Single inheritance is when there is a single parent class and a single child class which inherits the attributes and methods of the parent class. It reduces lines of code as instead of writing the same attributes and methods again in a new class, we can simply inherit them.

Code:

```
class Addition:
    def add(a,b):
        print("Sum:",a+b)
class Numbers(Addition):
    def getNumbers(self):
        c=float(input("Enter a number: "))
        d=float(input("Enter another number: "))
        Addition.add(c,d)
e=Numbers()
e.getNumbers()
```

Output(Screenshot):

+



Test Case: Any two (screenshot)

Conclusion: Hence, using single inheritance to take values from the user from method of child class and adding and printing the sum from method of parent class.

Experiment No: 4.6

Title: Write a program to create two base classes LU and ITM and one derived class. (Multiple inheritance).

Theory: Multiple inheritance is when a single child class inherits methods and attributes from multiple parent classes.

Code:

```
class LU:
    subjects=["AI/ML","AR/VR","Cloud Computing","Full Stack Developer"]
    teachers=["Sai Sir","Swapnil Sir","ABC","XYZ"]
    duration=["60 days","30 days","50 days","55 days"]
class ITM:
    subjects=["Hotel management","BTECH CSE","Design","Business"]
    teachers=["abc","Sumit sir","xyz","qwer"]
    duration=["90 days","120 days","60 days","30 days"]
```

```
class Btech(LU,ITM):
    print("LetsUpgrade courses are:\n")
    a=LU.subjects
    j=1
    for i in range(len(a)):
        print(j,LU.subjects[i],"by",LU.teachers[i],"for",LU.duration[i])
        j+=1
    c=[]
    d=int(input("How many subjects you want to select: "))
    if d>4:
        print("Invalid number of subjects")
    else:
        for i in range(d):
            e=int(input("Select your interested course: "))
            c.append(e)
    print("\nITM courses are:\n")
    j=1
    b=ITM.subjects
    for i in range(len(b)):
        print(j,ITM.subjects[i],"by",ITM.teachers[i],"for",ITM.duration[i])
        j+=1
    f=[]
    d=int(input("How many subjects you want to select: "))
    if d>4:
        print("Invalid number of subjects")
    else:
        for i in range(d):
            e=int(input("Select your interested course: "))
            f.append(e)
    j=1
    print("\nYour selected courses are:\n")
    for i in c:
        print(j,LU.subjects[i-1],"by",LU.teachers[i-1],"for",LU.duration[i-1])
        j+=1
    for i in f:
        print(j,ITM.subjects[i-1],"by",ITM.teachers[i-1],"for",ITM.duration[i-1])
        j+=1
obj=Btech()
```

Output(Screenshot):

+

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-6_LU_ITM.py
LetsUpgrade courses are:

1 AI/ML by Sai Sir for 60 days
2 AR/VR by Swapnil Sir for 30 days
3 Cloud Computing by ABC for 50 days
4 Full Stack Developer by XYZ for 55 days
How many subjects you want to select: 1
Select your interested course: 2

ITM courses are:

1 Hotel management by abc for 90 days
2 BTECH CSE by Sumit sir for 120 days
3 Design by xyz for 60 days
4 Business by qwer for 30 days
How many subjects you want to select: 1
Select your interested course: 2

Your selected courses are:

1 AR/VR by Swapnil Sir for 30 days
2 BTECH CSE by Sumit sir for 120 days
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Test Case: Any two (screenshot)

```
LetsUpgrade courses are:

1 AI/ML by Sai Sir for 60 days
2 AR/VR by Swapnil Sir for 30 days
3 Cloud Computing by ABC for 50 days
4 Full Stack Developer by XYZ for 55 days
How many subjects you want to select: 1
Select your interested course: 1

ITM courses are:

1 Hotel management by abc for 90 days
2 BTECH CSE by Sumit sir for 120 days
3 Design by xyz for 60 days
4 Business by qwer for 30 days
How many subjects you want to select: 1
Select your interested course: 1

Your selected courses are:

1 AI/ML by Sai Sir for 60 days
2 Hotel management by abc for 90 days
mahajanpiyush@Mahajans-MacBook-Air Mahajan_Piyush_Python_Lab %
```

Conclusion:

Hence, using multiple inheritance to show courses from multiple classes(LU and ITM) and print courses and their details which the user chooses.

Experiment No: 4.7

Title: Write a program to implement Multilevel inheritance, Grandfather->Father->Child to show property inheritance from grandfather to child.

Theory: Multilevel inheritance is when there is a parent class which has a child class which in turn has a child class of its own. The last child class inherits all the methods and attributes from its parent class as well as from the grandfather class.

Code:

```
class Grandfather:
    def __init__(self):
        self.name="ABC"
        self.inherit=10000
        self.purchase=1000
class Father(Grandfather):
    def __init__(self):
        super().__init__()
        self.name=" XYZ "+self.name
        self.inherit=self.inherit
        self.purchase=10000+self.purchase
class Child(Father):
    def __init__(self,name):
        super().__init__()
        self.name=name+self.name
        self.inherit=self.inherit
        self.purchase=self.purchase
        print("Hello",self.name)
        print("Inherited property: Rs",self.inherit)
        print("Purchased property: Rs",self.purchase)
        print("Total property:",self.inherit+self.purchase)
a=input("Enter your name: ")
```

```
obj=Child(a)
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/4-8_property_inhertance.py

Enter your name: Anusri
Hello Anusri XYZ ABC
Inherited property: Rs 10000
Purchased property: Rs 11000
Total property: 21000
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):

Conclusion:

Hence, using multilevel inheritance to take name from the user and calculate inherited and purchased property from grandfather and father for the child and their full name using methods and attributes from the grandfather and father class.

Experiment No: 4.8

Title: 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)

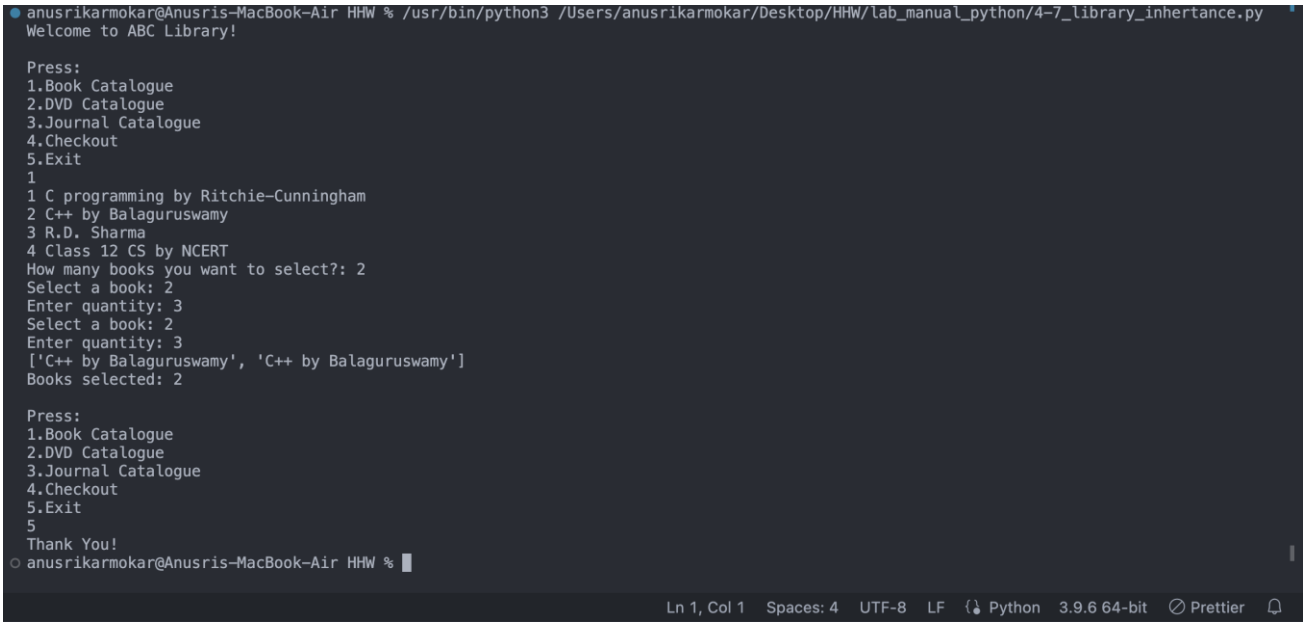
Theory: Method overriding is when there is a method of same name in both base class and derived class and when an object is created of derived class and method is called, derived class object is called and base class method is overridden.

Code:

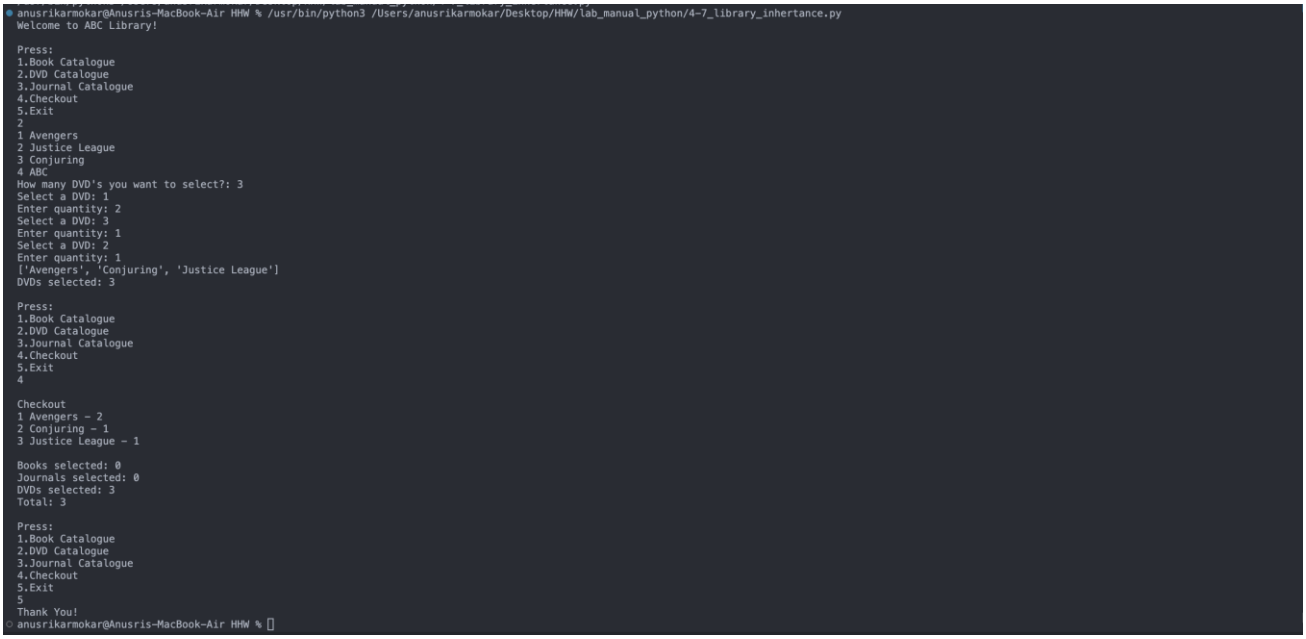
```
class Library_item:
        _bookCount=0
        _dvdCount=0
        _journalCount=0
        quantity=[]
        cart=[]
class Book(Library_item):
    def __init__(self):
        super().__init__()
        def check_out_book(self):
            d=int(input("How many books you want to select?: "))
            if d>4:
                print("Invalid choice")
            else:
```

```
            for i in range(d):
                c=int(input("Select a book: "))
                f=int(input("Enter quantity: "))
                Library_item.quantity.append(f)
                self.cart.append(a[c-1])
                Library_item._bookCount+=1
            print(self.cart)
            print("Books selected:",Library_item._bookCount)
        a=["C programming by Ritchie-Cunningham","C++ by Balaguruswamy","R.D. Sharma","Class
12 CS by NCERT"]
        b=1
        for i in range(len(a)):
            print(b,a[i])
            b+=1
        check_out_book(self)
class Dvd(Library_item):
    def __init__(self):
        super().__init__()
        def check_out_dvd(self):
            d=int(input("How many DVD's you want to select?: "))
            for i in range(d):
                if d>4:
                    print("Invalid choice")
                else:
                    c=int(input("Select a DVD: "))
                    f=int(input("Enter quantity: "))
                    Library_item.quantity.append(f)
                    self.cart.append(a[c-1])
                    Library_item._dvdCount+=1
            print(self.cart)
            print("DVDs selected:",Library_item._dvdCount)
        a=["Avengers","Justice League","Conjuring","ABC"]
        b=1
        for i in range(len(a)):
            print(b,a[i])
            b+=1
        check_out_dvd(self)
class Journal(Library_item):
    def __init__(self):
        super().__init__()
        def check_out_journal(self):
            d=int(input("How many Journal you want to select?: "))
            if d>4:
                print("Invalid choice")
            else:
                for i in range(d):
                    c=int(input("Select a journal: "))
                    f=int(input("Enter quantity: "))
                    Library_item.quantity.append(f)
                    self.cart.append(a[c-1])
```

```
                Library_item._journalCount+=1
            print(self.cart)
            print("Journals selected:",Library_item._journalCount)
        a=["A Journal","XYZ Journal","ABC Journal","QWERTY Journal"]
        b=1
        for i in range(len(a)):
            print(b,a[i])
            b+=1
        check_out_journal(self)
class Checkout(Library_item):
    def __init__(self):
        super().__init__()
        print("\nCheckout")
        j=1
        for i in range(len(self.cart)):
            print(j,self.cart[i],"-",self.quantity[i])
            j+=1
        print("\nBooks selected:",self._bookCount)
        print("Journals selected:",self._journalCount)
        print("DVDs selected:",self._dvdCount)
        print("Total:",self._bookCount+self._journalCount+self._dvdCount)
objs=list()
print("Welcome to ABC Library!")
for i in range(100):
    a=int(input("\nPress:\n1.Book Catalogue\n2.DVD Catalogue\n3.Journal
Catalogue\n4.Checkout\n5.Exit\n"))
    if a==1:
        objs.append(Book())
    elif a==2:
        objs.append(Dvd())
    elif a==3:
        objs.append(Journal())
    elif a==4:
        objs.append(Checkout())
    elif a==5:
        print("Thank You!")
        break
    else:
        print("Invalid choice")
        break
```



Output(Screenshot):



Test Case: Any two (screenshot)

Conclusion: Hence, using single inheritance and method overriding, made a library catalogue system having checkout system for books, movies, and journals.

Experiment No: 5.1

Title: Write a program to create my_module for addition of two numbers and import it in main script.

Theory: Module is a collection of functions. Its functions can be used in another program by importing the module.

Code:

```
import my_module
a=my_module.Addition()
b=float(input("Enter a number: "))
c=float(input("Enter another number: "))
a.add(b,c)
```

Output(Screenshot):

+

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/5-1_my_module_additional.p
y
Enter a number: 4
Enter another number: 4
Sum: 8.0
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/5-1_my_module_additional.p
y
Enter a number: 22
Enter another number: 33
Sum: 55.0
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/5-1_my_module_additional.p
y
Enter a number: 333
Enter another number: -1
Sum: 332.0
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Test Case: Any two (screenshot)

Conclusion: Hence, creating a module for addition of two values given by the user and printing it by importing the module in main program.

Experiment No: 5.2

Title: 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.

Theory:

Code:

```
from bank_module import ATM
b=ATM()
while True:
    a=int(input("Press 1 to deposit\nPress 2 to withdraw\nPress 3 to check balance\nPress 4 to
Exit\n"))
    if a==1:
```

```
        b.deposit()
    elif a==2:
        b.withdraw()
    elif a==3:
        b.check()
    elif a==4:
        print("Thank You")
        break
    else:
        print("Invalid choice\n")
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/5-2_atm.py
Welcome to ABC Bank
Enter your PIN: 1222
Enter your name: Anusri
Press 1 to deposit
Press 2 to withdraw
Press 3 to check balance
Press 4 to Exit
3
Balance: Rs 50000
Your Account Number XXXXXXXX1222

Press 1 to deposit
Press 2 to withdraw
Press 3 to check balance
Press 4 to Exit
4
Thank You
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot)

Test Case: Any two (screenshot)

Conclusion: Hence, creating a module for bank ATM with functions for deposit, withdraw, and checking bank balance and asking the user for deposit, withdraw or check bank balance and calling the user specified function using while loop.

Experiment No: 5.3

Title: 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.

Theory: A package is a collection of modules. It is a folder containing modules and __init__.py file to let python treat the folder as a package. Each module has methods for drive and start engine.

Code:

```
from cars.bmw import BMW
from cars.audi import Audi
from cars.nissan import Nissan

bmw_car = BMW(model="X5")
bmw_car.start_engine()
bmw_car.drive()
print()
audi_car = Audi(model="A4")
audi_car.start_engine()
audi_car.drive()
print()
nissan_car = Nissan(model="Altima")
nissan_car.start_engine()
nissan_car.drive()
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desk
top/HHW/lab_manual_python/cars/5-3_cars.py
BMW X5 engine started.
Driving the BMW X5.

Audi A4 engine started.
Driving the Audi A4.

Nissan Altima engine started.
Driving the Nissan Altima.
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):

Conclusion:

Hence, creating a package containing modules for different car models, with each module containing methods for each car model.

Experiment No: 6

Title: Write a program to implement Multithreading. Printing "Hello" with one thread & printing "Hi" with another thread.

Theory: Multithreading is used for multitasking as multiple processes are run in parallel at the same time. Each process is given a thread to run on and all the threads are joined together to create a main thread at the end of execution. Sleep is a function in time module which is used to suspend the execution of a thread for a specified number of seconds.

Code:

```
from threading import Thread
from time import sleep
def hi():
    for i in range(10):
        print("Hi")
        sleep(0.5)
def hello():
    for i in range(10):
        print("Hello")
        sleep(0.5)
t1=Thread(target=hi)
t2=Thread(target=hello)
t1.start()
t2.start()
t1.join()
t2.join()
print("End of execution")
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/6-1_multithreading.py
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hi
Hello
Hello
Hi
Hello
Hi
Hello
Hi
Hi
Hello
End of execution
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Output(Screenshot):

Conclusion: Hence, by using multithreading module in python to run multiple processes at once in parallel with each process getting their own threads to run on and joining them all at the end of execution of program. Also using sleep function from time module to let the processes on the threads run after a specified interval of time.

Experiment No: 7.1

Title: Write a program to use 'weather API 'and print temperature of any city, also print the sunrise and sunset times for the same humidity of that area.

Theory:

Requests module is used to connect a website with our program and an API key is required to get information from any website. API key is used for authentication of a user for getting any information from a website. Datetime module is used to convert Unix Epoch time to IST time for user convenience.

Code:

```
api_key="e8988573483b036cc946ddde037c0de6"
import requests
import datetime
city=input("Enter city: ")
response=requests.get(f"https://api.openweathermap.org/data/2.5/
weather?q={city}&APPID={api_key}&units=Metric")
a=response.json()
if 'message' in a:
print("City not Found!")
else:
print("\nCity:",city)
print("Temperature:",a['main']['temp'],"C")
```

```
print("Humidity:",a['main']['humidity'])
print("Sunrise(IST):",datetime.datetime.fromtimestamp(a['sys']
['sunrise']))
print("Sunset(IST):",datetime.datetime.fromtimestamp(a['sys']
['sunset']))
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-1_weather_api.py
/Users/anusrikarmokar/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:34: NotOpenSSLWarning: urllib3 v2 only supports Op
enSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Enter the city Name you want to know weather? melbourne

Name : melbourne                Weather : Clouds , broken clouds
Division  : Melbourne           Temp (°C) : 24.03
Feels as if (°C): 24.16         Humidity : 64
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-1_weather_api.py
```

Output(Screenshot):

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-1_weather_api.py
/Users/anusrikarmokar/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:34: NotOpenSSLWarning: urllib3 v2 only supports Op
enSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Enter the city Name you want to know weather? mumbai

Name : mumbai           Weather : Smoke , smoke
Division  : Konkan Division             Temp (°C) : 29
Feels as if (°C): 29.09         Humidity : 45
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-1_weather_api.py
/Users/anusrikarmokar/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:34: NotOpenSSLWarning: urllib3 v2 only supports Op
enSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Enter the city Name you want to know weather? london

Name : london           Weather : Clouds , broken clouds
Division  : London              Temp (°C) : 7.66
Feels as if (°C): 4.4           Humidity : 79
anusrikarmokar@Anusris-MacBook-Air HHW %
```

Test Case: Any two (screenshot)

Conclusion:

Hence, using API of openweather to get weather details of user given city and printing the weather details with the help of requests module and date time module to convert Unix epoch time to IST time.

Experiment No: 7.2

Title: Write a program to use the ʻAPI ʼof crypto currency.

Theory: Requests module is used to connect a website with our program and an API key is required to get information from any website. API key is used for authentication of a user for getting any information from a website.

Code:

```
import requests
api_id="CG-f2wXaQ33YUiT55jWb7AzSgvu"
while True:
coin=input("Enter cryptocoin: ")
response = requests.get(f"https://api.coingecko.com/api/v3/
simple/price?ids={coin}
&vs_currencies=usd,inr&x_cg_demo_api_key={api_id}")
a=response.json()
if coin in a:
print("\nCrypto:",coin)
print("Price:",a[coin]['usd'],"USD")
print("Price:",a[coin]['inr'],"INR")
else:
print("Invalid cryptocoin!")
b=input("Want to see more cryptocoins?(y/n): ")
if b.lower() == "n":
break
```

```
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-2_crypto_api.py
/Users/anusrikarmokar/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:34: NotOpenSSLWarning: urllib3 v2 only supports Op
enSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Enter cryptocoin: bitcoin
Press any key to see updates or "0" to exit8
₹ 3517026
Press any key to see updates or "0" to exit0
anusrikarmokar@Anusris-MacBook-Air HHW % /usr/bin/python3 /Users/anusrikarmokar/Desktop/HHW/lab_manual_python/7-2_crypto_api.py
```

Output(Screenshot):

Test Case: Any two (screenshot)

```
Enter cryptocoin: ethereum

Crypto: ethereum
Price: 2271.57 USD
Price: 188933 INR
Want to see more cryptocoins?(y/n): y
Enter cryptocoin: abc
Invalid cryptocoin!
Want to see more cryptocoins?(y/n): n
```

Conclusion:

Hence, using API of CoinGecko to get price of cryptocurrency given by the user and printing INR and USD price and asking for more crypto till the user chooses the option to terminate the program using while loop.