***Artificial Intelligence Lab***

***CSL 411***

***Lab Journal 2***

****

**Name:Rida sajjad**

**Enrollment:01-134152-063**

**BSCS-7B**

**Department of Computer Science**

**BAHRIA UNIVERSITY, ISLAMABAD**

**Lab # 2-1: Classes, Inheritance, and Modules in Python**

**Lab # 2-2: Lists, Tuples, Set, Dictionary, and Generator in Python**

**Objectives:**

To learn about Object Oriented Programming in python.

To learn about different data structures in python and how to use them.

**Tools Used:**

IDLE (Python 3.4 GUI Python)

**Submission Date:**

**Evaluation: Signatures of Lab Engineer:**

**Task # 2A-1:**

**Procedure/Program:**

**Base class :**

import math

class basic\_cal:

def \_\_init\_\_(self,a,b):

self.x=a

self.y=b

def add(self):

return self.x+self.y

def mult(self):

return self.x\*self.y

def div(self):

return self.x/self.y

**Derived class:**

class s\_cal(basic\_cal):

def \_\_init\_\_(self,x,y):

self.x=x

self.y=y

def fact(self):

print(math.factorial(self.x))

def power(self):

print (math.pow(self.x,self.y))

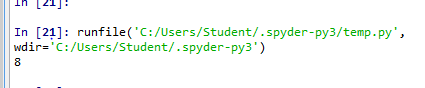
def logn(self):

return math.log1p(self.x)

a=basic\_cal(3,5)

print(a.add())

**Result/Output:**



**Task # 2A-2:**

**Procedure/Program:**

**Module.class:**

import math

class basic\_cal:

def \_\_init\_\_(self,a,b):

self.x=a

self.y=b

def add(self):

return self.x+self.y

def mult(self):

return self.x\*self.y

def div(self):

return self.x/self**.y**

**anotherfile:**

import math

import meramodule

class s\_cal(meramodule.basic\_cal):

def \_\_init\_\_(self,x,y):

self.x=x

self.y=y

def fact(self):

print(math.factorial(self.x))

def power(self):

print (math.pow(self.x,self.y))

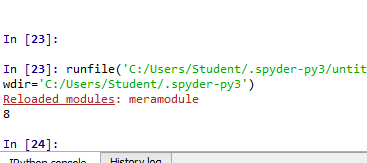
def logn(self):

return math.log1p(self.x)

a=s\_cal(3,5)

print(a.add())

**Result/Output:**



**Task # 2B-1:**

**Procedure/Program:**

lower = int(input("Enter lower range: "))

upper = int(input("Enter upper range: "))

print("Prime numbers between",lower,"and",upper,"are:")

for num in range(lower,upper + 1):

# prime numbers are greater than 1

if num > 1:

for i in range(2,num):

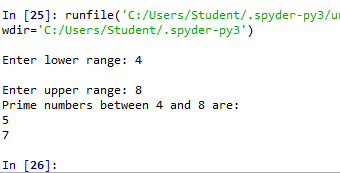
if (num % i) == 0:

break

else:

print(num)

**Result/Output:**



**Task # 2B-2:**

**Procedure/Program: Result/Output:**

terms = int(input("How many terms? "))

n1 = 0

n2 = 1

count = 0

if terms <= 0:

print("Please enter a positive integer")

elif terms == 1:

print("Fibonacci sequence upto",terms,":")

print(n1)

else:

print("Fibonacci sequence upto",terms,":")

while count < terms:

print(n1,end=' , ')

nth = n1 + n2

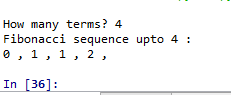
# update values

n1 = n2

n2 = nth

count += 1

**Analysis/Conclusion:**



In above task we learn how to make fibseries in python

**Task # 2B-3:**

**Procedure/Program:**

**p = 'yes'**

**word= input('Enter a word:')**

**if len(word) > 0 and word.isalpha():**

**words = word.lower()**

**first = words[0]**

**if first == ('a' or 'e' or 'i' or 'o' or 'u'):**

**new\_word = words + p**

**print (new\_word)**

**else:**

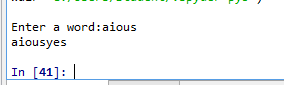
**new\_word = words[1:] + first + p**

**print (new\_word)**

**else:**

**print ('empty')**

**Result/Output:**



**Task # 2B-4:**

**Procedure/Program:**

import math

import meramodule

class s\_cal(meramodule.basic\_cal):

def \_\_init\_\_(self,x,y):

self.x=x

self.y=y

def fact(self): print(math.factorial(self.x))

def power(self): print (math.pow(self.x,self.y))

def logn(self):return math.log1p(self.x)

def sin(self):return math.sin(self.x)

def cos(self): return math.cos(self.x)

def tan(self):return math.tan(self.x)

def sqr(self): return math.sqrt(self.x)

a=s\_cal(90,60)

print(a.sin())

print(a.cos())

print(a.tan())

print(a.sqr())

**Result/Output:**

