**CYCLE IV**

**Program 01**

**Aim:-**

Write a program to print the Fibonacci series using recursion.

**Source code:-**

def fibonacci(n):

if n<=1:

return n

else:

return fibonacci(n-1)+fibonacci(n-2)

limit=int(input("Enter the number of terms: "))

if limit<=0:

print("Plaese Enter a Positive Integer:")

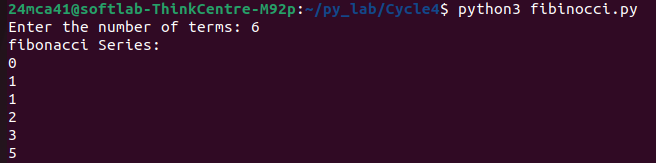
else:

print("fibonacci Series:")

for i in range(limit):

print(fibonacci(i))

**Output:-**



**Program 02**

**Aim:-**

Write the to implement a menu-driven calculator. Use separate functions for the

different operations.

**Source code:-**

def add(x,y):

return x+y

def subtract(x,y):

return x-y

def Multiply(x,y):

return x\*y

def Divide(x,y):

if y==0:

return "Error!! Division by zero"

else:

return x/y

def calculator():

while True:

print("\nSelect an operation:\n1.Add\n2.Subtract\n3.Multiply\n4.Divition\n5.exit\n")

choice=int(input("Enter your choice:"))

if choice==5:

print("Exiting the calculator!")

return

else:

num1=float(input("Enter First Number: "))

num2=float(input("Enter Second Number: "))

if choice==1:

print(f"result:{num1}+{num2}={add(num1,num2)}")

elif choice==2:

print(f"Result:{num1}-{num2}={subtract(num1,num2)}")

elif choice==3:

print(f"Result:{num1}\*{num2}={Multiply(num1,num2)}")

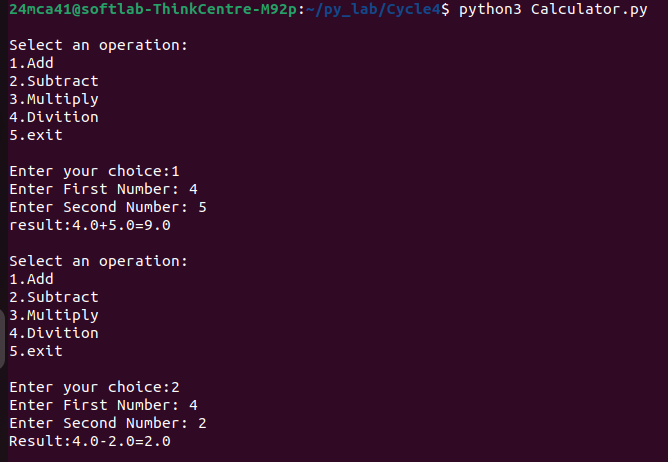
elif choice==4:

print(f"Result:{num1}/{num2}={Divide(num1,num2)}")

else:

print("Invalid Input!!")

calculator()



**Program 03**

**Aim:-**

Write a program to print the nth prime number.[Use function to check whether a number is prime or not]

**Source code:-**

def is\_prime(num):

if num<=1:

return False

for i in range(2,int(num \*\* 0.5)+1):

if num%i==0:

return True

def nth\_prime(n):

count=0

number=2

while True:

if is\_prime(number):

count+=1

if count==n:

return number

number+=1

n=int(input("Enter the Positon of Prime number: "))

if n<=0:

print("Invalid Input!!")

else:

print(f"the {n} the prime number is:{nth\_prime(n)}")

**Output:-**

****

**Program 04**

**Aim:-**

Write lambda functions to find the area of square, rectangle and triangle.

**Source code:-**

area\_square=lambda S\_side:S\_side \*\*2

area\_rectangle=lambda rect\_length,rect\_width:rect\_length \* rect\_width

area\_triangle=lambda t\_base,t\_height:0.5 \* t\_base \* t\_height

S\_side=int(input("Enter Square side: "))

print("Area of Square: ",area\_square(S\_side))

rect\_length=int(input("Enter Rectangle length: "))

rect\_width=int(input("Enter Rectangle width: "))

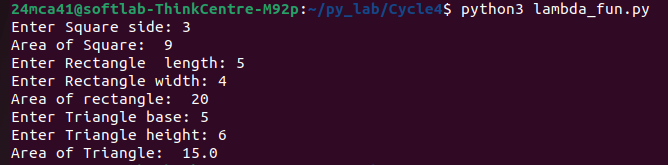
print("Area of rectangle: ",area\_rectangle(rect\_length,rect\_width))

t\_base=int(input("Enter Triangle base: "))

t\_height=int(input("Enter Triangle height: "))

print("Area of Triangle: ",area\_triangle(t\_base,t\_height))

**Output:-**



**Program 05**

**Aim:-**

Write a program to display powers of 2 using anonymous function. [ Hint use map and

lambda function)

**Source code:-**

twox=lambda x:2\*\*x

terms=list(map(int,input("Enter the list of item: ").split(" ")))

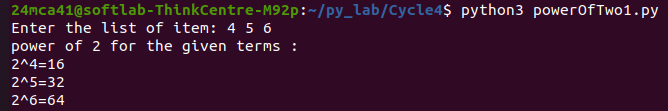
power\_of\_2=list(map(twox,terms))

print("power of 2 for the given terms :")

for i in range(len(terms)):

print(f"2^{terms[i]}={power\_of\_2[i]}")

**Output:-**



**Program 06**

**Aim:-**

Write a program to display multiples of 3 using anonymous function. [ Hint use filter and

lambda function)

**Source code:-**

terms = int(input("Enter the number of terms: "))

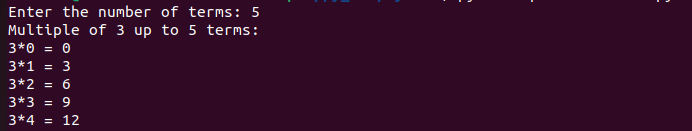
indices = range(terms)

Multiple\_of\_3 = list(filter(lambda x: True, map(lambda i: 3\*i, indices)))

print(f"Multiple of 3 up to {terms} terms:")

for i in range(terms):

print(f"3\*{i} = {Multiple\_of\_3[i]}")

****

**Program 07**

**Aim:-**

Write a program to sum the series 1/1! + 4/2! + 27/3! + ….. + nth term. [ Hint

Use a function to find the factorial of a number]

**Source code:-**

def factorial(num):

if num==1 or num==0:

return 1

else:

fact=1

for i in range(2,num+1):

fact=fact\*i

return fact

def nth\_def factorial(num):

if num==1 or num==0:

return 1

else:

fact=1

for i in range(2,num+1):

fact=fact\*i

return fact

def nth\_term(n):

return (n\*\*n)/(factorial(n))

def sum\_series(n):

total\_sum=0

for i in range(1,n+1):

total\_sum+=nth\_term(i)

return total\_sum

n=int(input("Enter the number of terms: "))

print(f"The sum of the Series upto {n} term is : {sum\_series(n)}")term(n):

return (n\*\*n)/(factorial(n))

def sum\_series(n):

total\_sum=0

for i in range(1,n+1):

total\_sum+=nth\_term(i)

return total\_sum

n=int(input("Enter the number of terms: "))

print(f"The sum of the Series upto {n} term is : {sum\_series(n)}")

**Output:-**



**Program 08**

**Aim:-**

Write a function called compare which takes two strings S1 and S2 and an integer n as

arguments. The function should return True if the first n characters of both the strings

are the same else the function should return False.

**Source code:-**

def Compare(n,s1,s2):

if s1[:n]==s2[:n]:

return True

else:

return False

s1=input("Enter first String: ")

s2=input("Enter second String: ")

n=int(input("Enter n value: "))

r=Compare(n,s1,s2)

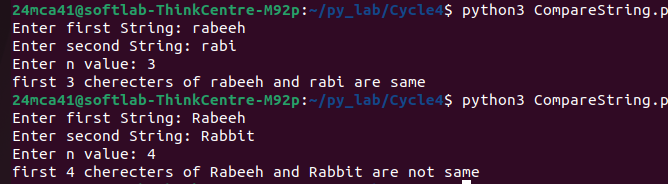
if r==True:

print(f"first {n} cherecters of {s1} and {s2} are same")

elif r==False:

print(f"first {n} cherecters of {s1} and {s2} are not same")

**Output:-**

****

**Program 09**

**Aim:-**

Write a program to add variable length integer arguments passed to the function. [Also

demo the use of docstrings]

**Source code:-**

def add\_numbers(\*args):

"""

Adds a variable number of integer arguments.

parameters:

\*args:A variable length list of Integers to be added.

returns:

int:the sum of all the integers passed as argumens.

"""

if not all(isinstance(arg,int)for arg in args):

raise valueError("All arguments must be integers!!")

return sum(args)

print("sum of 1,2,3:",add\_numbers(1,2,3))

print("sum of 10,20,30,40:",add\_numbers(10,20,30,40))

**Output:-**



**Program 10**

**Aim:-**

Write a program using functions to implement these formulae for permutations and

combinations.

The Number of permutations of n objects taken r at a time: p(n, r) = n!/(n − r)!.

The Number of combinations of n objects taken r at a time is: c(n, r) = n!/(r! ∗ (n − r)!)

**Source code:-**

def factorial(num):

if num==1 or num==0:

return 1

else:

fact=1

for i in range(2,num+1):

fact=fact\*i

return fact

def Permutation(n,r):

return factorial(n) // factorial(n-r)

def Combination(n,r):

return factorial(n) // (factorial(r) \* factorial(n-r))

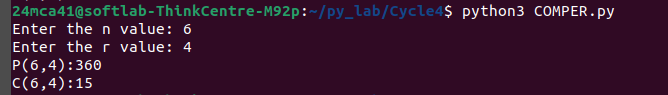
n=int(input("Enter the n value: "))

r=int(input("Enter the r value: "))

print(f"P({n},{r}):{Permutation(n,r)}")

print(f"C({n},{r}):{Combination(n,r)}")

**Output:-**

****