**CYCLE III**

**Program1:-**

**Aim:-**

Write a program to find the factorial of a number

**Source code:-**

num=int(input("Enter the number to find Factorial: "))

fact=1

if num<0:

print("inalid input!")

else:

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

fact=fact\*i

print("factorial of ",num,"=",fact)



**Program2:-**

**Aim:-**

Generate Fibonacci series of N terms

**Source code:-**

n=int(input("Enter the limit of series: "))

a=0

b=1

count=0

if n<0:

print("Invalid input")

elif n==1:

print(a,"\t")

else:

while count<n:

print(a,end=" ")

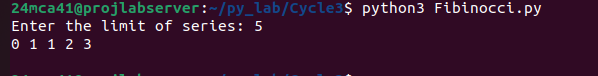
sum=a+b

a=b

b=sum

count=count+1

print("\n")

****

**Program3:-**

**Aim:-**

Write a program to find the sum of all items in a list. [Using for loop]

**Source code:-**

size=int(input("Enter the size of the list: "))

list=[]

sum=0

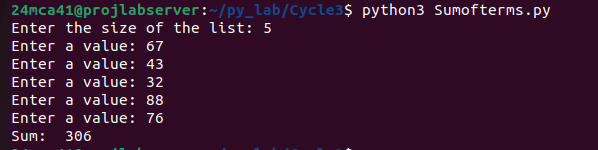
for i in range(size):

val=int(input("Enter a value: "))

list.append(val)

sum=sum+val

print("Sum: ",sum)

****

**Program4:-**

**Aim:-**

Generate a list of four digit numbers in a given range with all their digits even and

the number is a perfect square

**Source code:-**

import math

def even\_square(start,end):

result=[]

L\_bound=math.ceil(math.sqrt(start))

U\_bound=math.floor(math.sqrt(end))

for i in range(L\_bound,U\_bound):

square=i\*\*2

if all(int(digit)%2==0 for digit in str(square)):

result.append(square)

return result

start\_range=1000

end\_range=9999

even\_digit=even\_square(start\_range,end\_range)

print("four digit perfect squares with all even digits: ",even\_digit)

****

**Program5:-**

**Aim:-**

Write a program using a for loop to print the multiplication table of n, where n is

entered by the user.

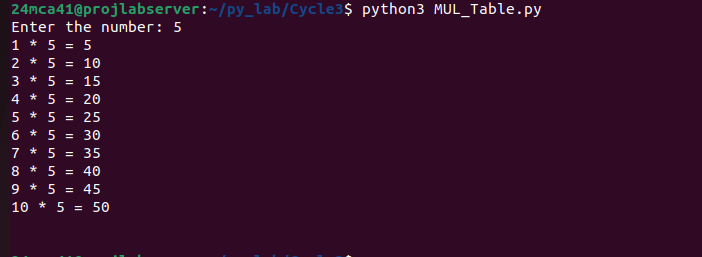
**Source code:-**

num=int(input("Enter the number: "))

for i in range(1,11):

print(i,"\*",num,"=",i\*num)

print("\n")

****

**Program6:-**

**Aim:-**

Write a program to display alternate prime numbers till N (obtain N from the user).

**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 False

return True

def Alt\_primes(n):

primes=[]

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

if is\_prime(i):

primes.append(i)

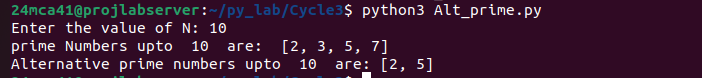
alt\_primes=primes[::2]

print("prime Numbers upto ",N," are: ",primes)

return alt\_primes

N=int(input("Enter the value of N: "))

print("Alternative prime numbers upto ",N," are:",Alt\_primes(N))

****

**Program7:-**

**Aim:-**

Write a program to compute and display the sum of all integers that are divisible

by 6 but not by 4, and that lie below a user-given upper limit.

**Source code:-**

Upper\_limit=int(input("Enter the Upper Limit: "))

sum=0

for i in range(1,Upper\_limit):

if i%6==0 and i%4!=0:

sum=sum+i

print("Sum= ",sum)

****

**Program8:-**

**Aim:-**

Calculate the sum of the digits of each number within a specified range (from 1 to

a user-defined upper limit). Print the sum only if it is prime.

**Source code:-**

Upper\_limit=int(input("Enter the Upper limit: "))

print("prime number are:")

for i in range(1,Upper\_limit):

temp=i

sum=0

while temp>0:

digit=temp%10

temp=temp//10

sum=sum+digit

flag=0

if sum<=1:

continue

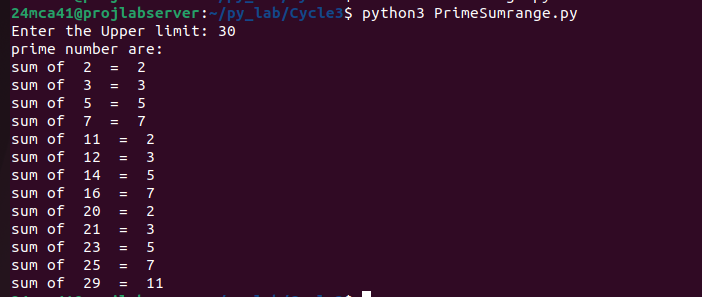
for j in range(2,sum):

if sum%j==0:

flag=1

if flag==0:

print("sum of ",i," = ",sum)



**Program9:-**

**Aim:-**

A number is input through the keyboard. Write a program to determine if it’s

palindromic

**Source code:-**

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

temp=n

reverse=0

while temp>0:

digit=temp%10

temp=temp//10

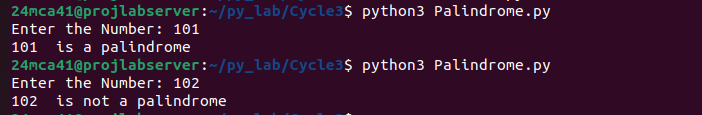
reverse=reverse \* 10 +digit

if n==reverse:

print(n," is a palindrome")

else:

print(n," is not a palindrome")

****

**Program10:-**

**Aim:-**

Write a program to generate all factors of a number. [use while loop]

**Source code:-**

n=int(input("Enter a Number: "))

print("Factors of ",n," = ",end=" ")

count=1

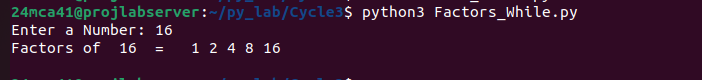
while count<=n:

if n%count==0:

print(count,end=" ")

count=count+1

print("\n")

****

**Program11:-**

**Aim:-**

Write a program to find whether the given number is an Armstrong number or

not. [use while loop]

**Source code:-**

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

temp=n

digits=0

while temp>0:

temp=temp//10

digits=digits+1

temp=n

arm=0

while temp>0:

arm=arm+(temp%10)\*\*digits

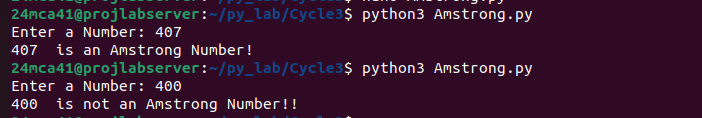
temp=temp//10

if arm==n:

print(n," is an Amstrong Number!")

else:

print(n," is not an Amstrong Number!!")

****

**Program12:-**

**Aim:-**

Display the given pyramid with the step number accepted from the user. Eg: N=4

1

2 4

3 6 9

4 8 12 16

**Source code:-**

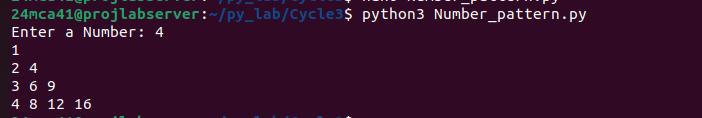
n=int(input("Enter a Number: "))

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

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

print(i\*j,end=" ")

print()

****

**Program13:-**

**Aim:-**

Construct following pattern using nested loop

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

**Source code:-**

n=int(input("Enter Number of Rows: "))

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

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

print('\*', end=" ")

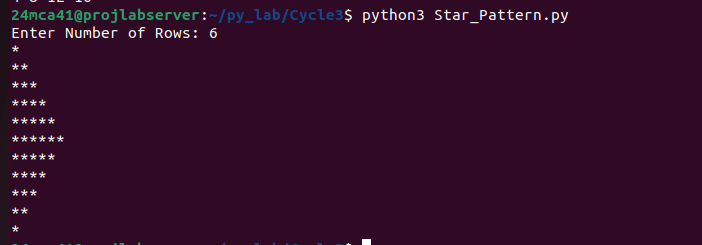
print()

for i in range(n-1,0,-1):

for j in range(i-1,0,-1):

print('\*', end=" ")

print()

****