**DATA SCIENCE LAB**

**CYCLE 1**

**ROSMI REJI**

**S3 MCA**

**ROLL NO 47**

**1.Program to Print all non-Prime Numbers in an Interval**

CODE:

#Take the input from the user:

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

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

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

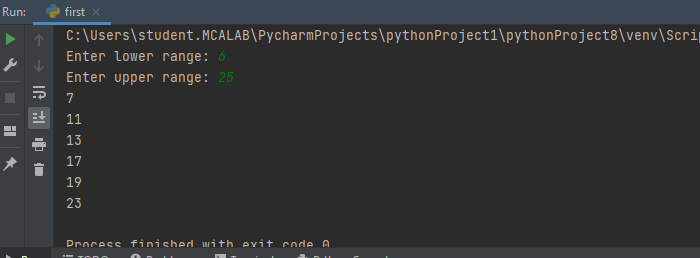
if num > 1:

for i in range(2,num):

if (num % i) == 0:

break

else: print(num)



**2. Program to print the first N Fibonacci numbers.**

CODE:

n = int (input("Enter the number of terms needed in the Fibonacci series: "))

if (n<0):

print ("Enter a positive number")

else:

f1, f2 = 0, 1

if n == 1:

print (f1)

elif n == 2:

print (f1,f2)

else:

print (f1,f2, end = ' ')

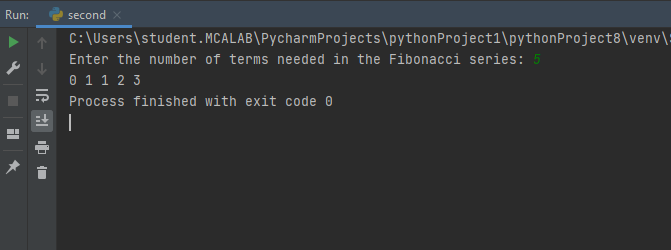
for i in range (3, n+1):

f3 = f1 + f2

print (f3, end = ' ')

f1 = f2

f2 = f3



**3. Given sides of a triangle, write a program to check whether given triangle is an isosceles, equilateral or scalene.**

CODE:

print("Input lengths of the triangle sides: ")

x = int(input("x: "))

y = int(input("y: "))

z = int(input("z: "))

if x == y == z:

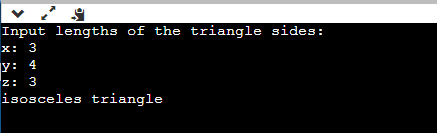
print("Equilateral triangle")

elif x==y or y==z or z==x:

print("isosceles triangle")

else:

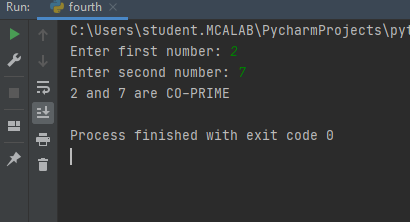
print("Scalene triangle")



**4. Program to check whether given pair of number is coprime**

CODE: def are\_coprime(a,b):  
  
 hcf = 1  
  
 for i in range(1, a+1):  
 if a%i==0 and b%i==0:  
 hcf = i  
  
 return hcf == 1  
  
# Reading two numbers  
first = int(input('Enter first number: '))  
second = int(input('Enter second number: '))

if are\_coprime(first, second):  
 print('%d and %d are CO-PRIME' %(first, second))  
else:  
 print('%d and %d are NOT CO-PRIME' %(first, second))



5. Program to find the roots of a quadratic equation(rounded to 2 decimal places)

CODE:

from math import sqrt

print("Quadratic function : (a \* x^2) + b\*x + c")

a = float(input("a: "))

b = float(input("b: "))

c = float(input("c: "))

r = b\*\*2 - 4\*a\*c

if r > 0:

num\_roots = 2

x1 = (((-b) + sqrt(r))/(2\*a))

x2 = (((-b) - sqrt(r))/(2\*a))

print("There are 2 roots: %f and %f" % (x1, x2))

elif r == 0:

num\_roots = 1

x = (-b) / 2\*a

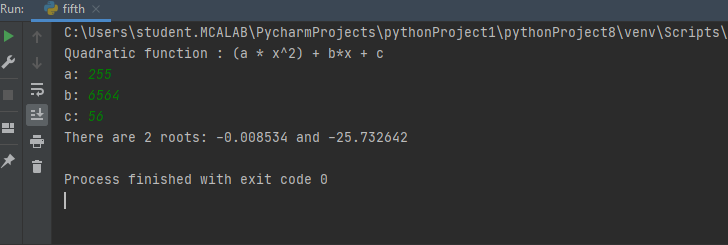
print("There is one root: ", x)

else:

num\_roots = 0

print("No roots, discriminant < 0.")

exit()



6. Program to check whether a given number is perfect number or not(sum of factors=number)

CODE:

n = int(input("Enter any number: "))

sum1 = 0

for i in range(1, n):

if(n % i == 0):

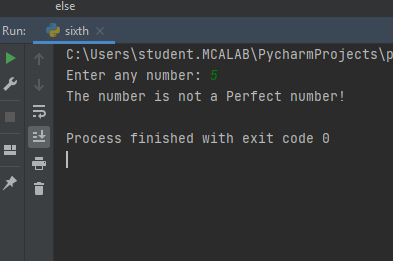
sum1 = sum1 + i

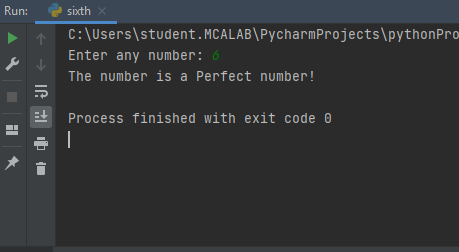
if (sum1 == n):

print("The number is a Perfect number!")

else:

print("The number is not a Perfect number!")





7. Program to display amstrong numbers upto 1000

CODE:

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

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

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

**sum = 0**

**temp = num**

**while temp > 0:**

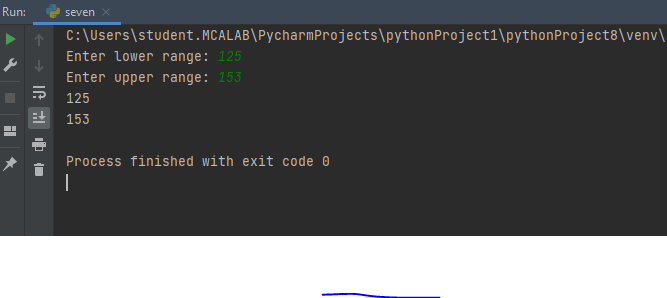
**digit = temp % 10**

**sum += digit \*\* 3**

**temp //= 10**

**if num == sum:**

**print(num)**



8. Store and display the days of a week as a List, Tuple, Dictionary, Set. Also demonstrate different ways to store values in each of them. Display its type also.

CODE:

list = ["Sun","Mon","Tue","Wed","Thu","Fri","Sat"]

print(type(list))

print(list)

tuple = ("Sun","Mon","Tue","Wed","Thu","Fri","Sat")

print(type(tuple))

print(tuple)

set = {"Sun","Mon","Tue","Wed","Thu","Fri","Sat"}

print(type(set))

print(set)

dict = {

    "d1" : "Sun",

    "d2" : "Mon",

    "d3" : "Tue",

    "d4" : "Wed",

    "d5" : "Thu",

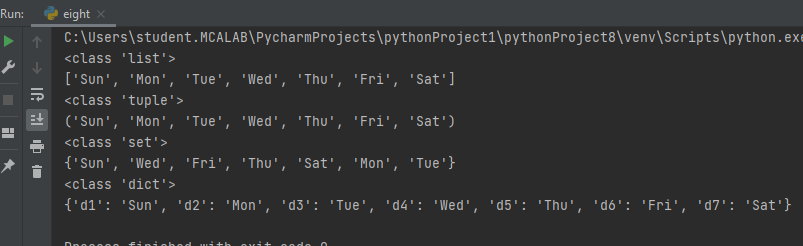
    "d6" : "Fri",

    "d7" : "Sat"

}

print(type(dict))

print(dict)



9. Write a program to add elements of given 2 lists

CODE:

**# initialize the Python lists**

**lt1 = [5, 10, 15, 20, 25, 30]**

**lt2 = [2, 4, 6, 8, 10, 12]**

**# print the original list element**

**print ( " Python Original list 1: " + str (lt1))**

**print ( "Python Original list 2: " + str (lt2))**

**# use naive method to add two list.**

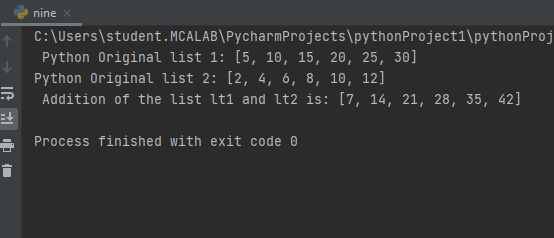
**res\_lt = [] # declaration of the list**

**for x in range (0, len (lt1)):**

**res\_lt.append( lt1[x] + lt2[x])**

**# Display the sum of two list in Python**

**print ( " Addition of the list lt1 and lt2 is: " + str (res\_lt))**



10. Write a program to find the sum of 2 matrices using nested List.

CODE:

**matOne = []**

**print("Enter Elements for First Matrix: ")**

**for i in range(3):**

**matOne.append([])**

**for j in range(3):**

**num = int(input())**

**matOne[i].append(num)**

**matTwo = []**

**print("Enter Elements for Second Matrix: ")**

**for i in range(3):**

**matTwo.append([])**

**for j in range(3):**

**num = int(input())**

**matTwo[i].append(num)**

**matThree = []**

**for i in range(3):**

**matThree.append([])**

**for j in range(3):**

**matThree[i].append(matOne[i][j]+matTwo[i][j])**

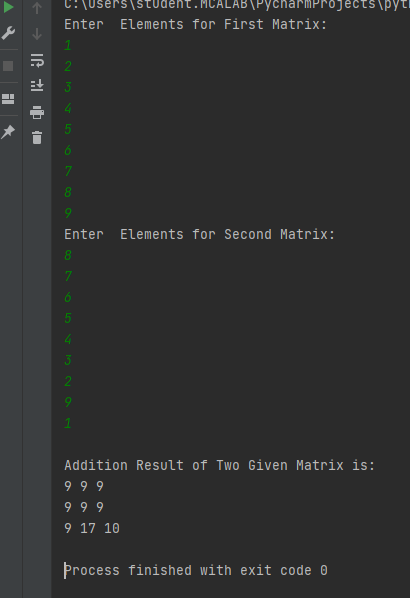
**print("\nAddition Result of Two Given Matrix is:")**

**for i in range(3):**

**for j in range(3):**

**print(matThree[i][j], end=" ")**

**print()**



11. Write a program to perform bubble sort on a given set of elements.

CODE: **# Creating a bubble sort function**

**def bubble\_sort(list1):**

**# Outer loop for traverse the entire list**

**for i in range(0,len(list1)-1):**

**for j in range(len(list1)-1):**

**if(list1[j]>list1[j+1]):**

**temp = list1[j]**

**list1[j] = list1[j+1]**

**list1[j+1] = temp**

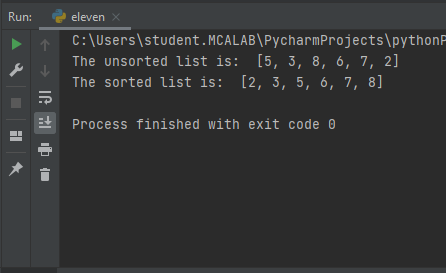
**return list1**

**list1 = [5, 3, 8, 6, 7, 2]**

**print("The unsorted list is: ", list1)**

**# Calling the bubble sort function**

**print("The sorted list is: ", bubble\_sort(list1))**



12. Program to find the count of each vowel in a string(use dictionary)

CODE:

**def Check\_Vow(string, vowels):**

**# The term "casefold" has been used to refer to a method of ignoring cases.**

**string = string.casefold()**

**count = {}.fromkeys(vowels, 0)**

**# To count the vowels**

**for character in string:**

**if character in count:**

**count[character] += 1**

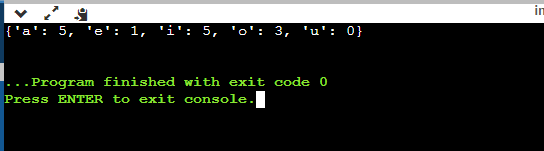
**return count**

**# Driver Code**

**vowels = 'aeiou'**

**string = "Hi, I AM ROSMI REJI AND I AM FROM KOTTAYAM"**

**print (Check\_Vow(string, vowels))**

****

13. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is

Positive

CODE:

code: *def* repeat\_times(n):

s = 0

n\_str = *str*(n)

while (n > 0):

n -= *sum*([*int*(i) for i in *list*(n\_str)])

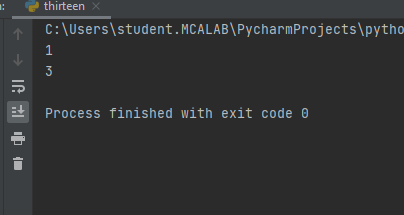
n\_str = *list*(*str*(n))

s += 1

return s

print(repeat\_times(9))

print(repeat\_times(21))



14. Write a Python program that accepts a 10 digit mobile number, and find the digits which are absent in a given mobile number

def absent\_digits(n):

  all\_nums = set([0,1,2,3,4,5,6,7,8,9])

  n = set([int(i) for i in n])

  n = n.symmetric\_difference(all\_nums)

  n = sorted(n)

  return n

print(absent\_digits([9,5,2,6,0,1,4,6,8,4]))

