**DATA SCEINCE & MACHINE LEARNING**

**LAB CYCLE 1**

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

**PROGRAM:**

first=int(input("Enter the First Limit:"))

last=int(input("Enter the last limit"))

for num in range(first,last + 1):

if num > 1:

for i in range(2,num):

if(num % i == 0):

print(num)

break

**OUTPUT:**



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

**PROGRAM:**

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):

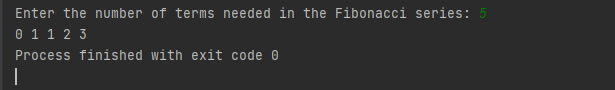
f3 = f1 + f2

print (f3, end = ' ')

f1 = f2

f2 = f3

**OUTPUT:**



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

**PROGRAM:**

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")

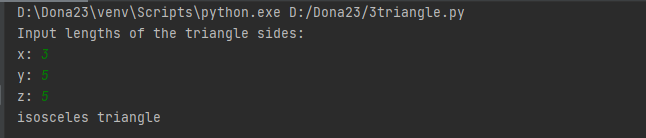
elif x==y or y==z or z==x:

print("isosceles triangle")

else:

print("Scalene triangle")

**OUTPUT:**



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

**PROGRAM:**

def gcd(p,q):

# Create the gcd of two positive integers.

while q != 0:

p, q = q, p%q

return p

def is\_coprime(x, y):

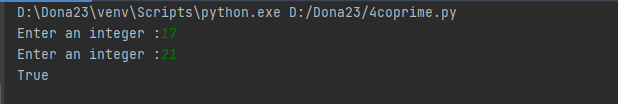
return gcd(x, y) == 1

a = (int(input("Enter an integer :")))

b = (int(input("Enter an integer :")))

print(is\_coprime(a , b))

**OUTPUT:**



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

**PROGRAM:**

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

import cmath

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

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

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

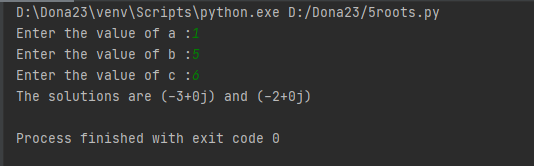
d = (b\*\*2) - (4\*a\*c)

sol1 = (-b-cmath.sqrt(d))/(2\*a)

sol2 = (-b+cmath.sqrt(d))/(2\*a)

print('The solutions are {0} and {1}'.format(sol1,sol2))

**OUTPUT:**



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

**PROGRAM:**

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

n = int(input(" Please Enter any Number: "))

Sum = 0

for i in range(1, n):

if(n % i == 0):

Sum = Sum + i

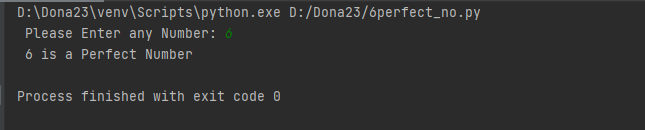
if (Sum == n):

print(" %d is a Perfect Number" %n)

else:

print(" %d is not a Perfect Number" %n)

**OUTPUT:**



1. **Program to display amstrong numbers upto 1000**

**PROGRAM:**

#Program to display amstrong numbers upto 1000

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

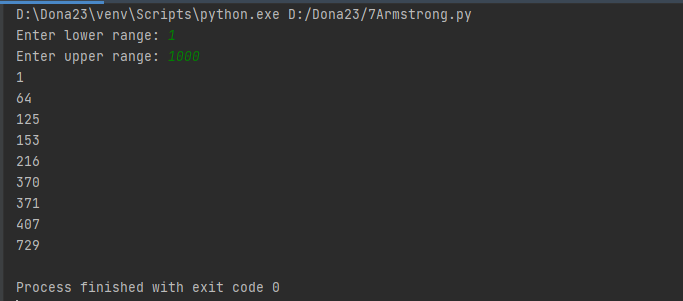
sum += digit \*\* 3

temp //= 10

if num == sum:

print(num)

**OUTPUT:**



1. **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.**

**PROGRAM:**

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",

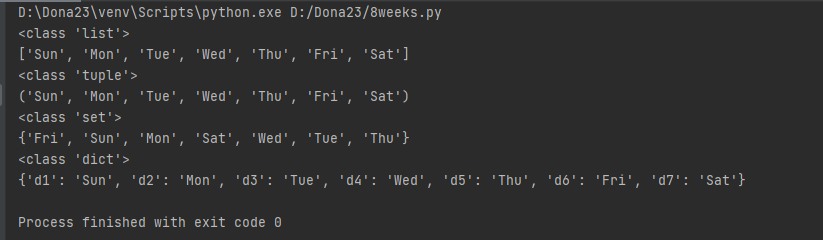
"d7" : "Sat"

}

print(type(dict))

print(dict)

**OUTPUT:**



1. **Write a program to add elements of given 2 lists**

**PROGRAM:**

l1 = []

print("Enter number of elements in list1 : ")

n1 = int(input())

print("Enter elements in list1 : ")

for i in range(0, n1):

ele1 = int(input())

l1.append(ele1)

print("List 1 : ", l1)

l2 = []

print("Enter number of elements in list2 : ")

n2 = int(input())

print("Enter elements in list2 : ")

for i in range(0, n2):

ele2 = int(input())

l2.append(ele2)

print("List 2 : ", l2)

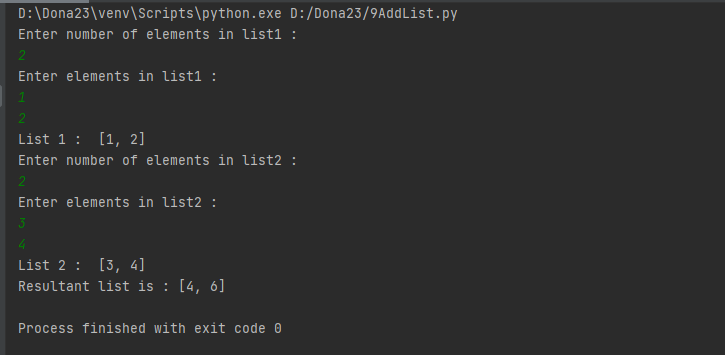
result = []

for i in range(0, len(l1)):

result.append(l1[i] + l2[i])

print("Resultant list is : " + str(result))

**OUTPUT:**



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

**PROGRAM:**

a = []

a1 = [1, 2, 3]

a.append(a1)

a2 = [4, 5, 6]

a.append(a2)

print(a)

b =[]

b1 = [1, 1, 3]

b.append(b1)

b2 = [2, 4, 2]

b.append(b2)

print(b)

result = [[0,0,0],

[0,0,0]]

print("Resultant matrix : ")

for i in range(len(a)):

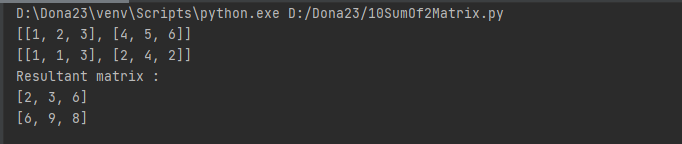
for j in range(len(a[0])):

result[i][j] = a[i][j] + b[i][j]

for r in result:

print(r)

**OUTPUT:**



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

**PROGRAM:**

def bubbleSort(arr):

for i in range(n - 1):

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

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

arr = []

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

print("Enter elements :")

for i in range(0,n):

arr.append(int(input()))

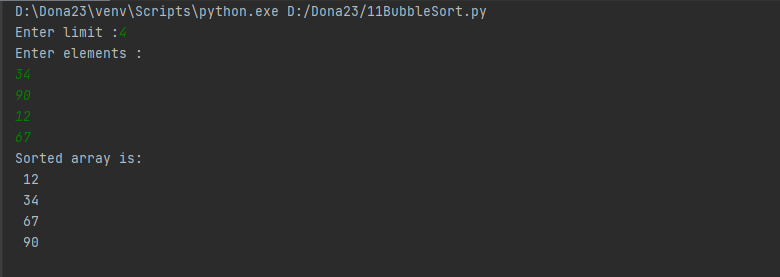
bubbleSort(arr)

print("Sorted array is:")

for i in range(len(arr)):

print("% d" % arr[i])

**OUTPUT:**



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

**PROGRAM:**

string = input("Enter a string :")

lowercase = string.lower()

vowel\_counts = {}

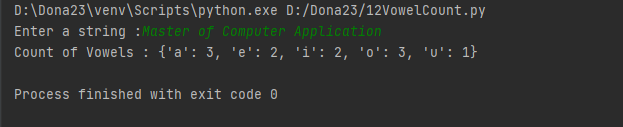
for vowel in "aeiou":

count = lowercase.count(vowel)

vowel\_counts[vowel] = count

print("Count of Vowels :", vowel\_counts)

**OUTPUT:**



1. **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**

**PROGRAM:**

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))

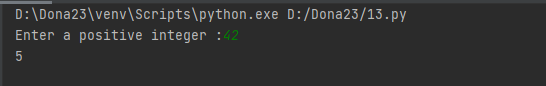
s += 1

return s

n=int(input("Enter a positive integer :"))

print(repeat\_times(n))

**OUTPUT:**



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

**PROGRAM:**

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

mobile = input('Please enter a mobile number: ' )

all = '0123456789'

print('Missing digits are ', set(all) - set(mobile))

**OUTPUT:**

