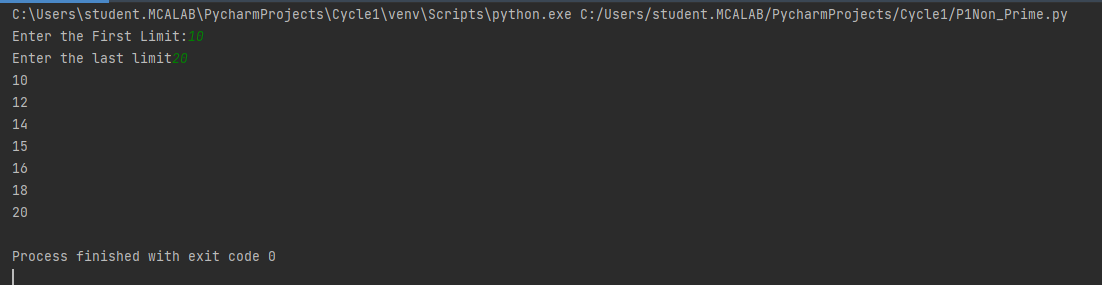
**DATA SCEINCE & MACHINE LEARNING:**

**LAB CYCLE 1**

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

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

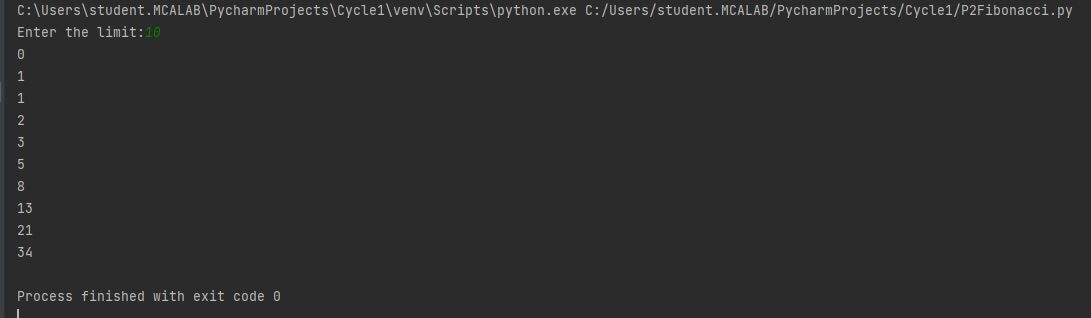
**Output:**



1. Program to print the first N Fibonacci numbers.

n = int (input("Enter the limit:"))  
n1=0  
n2=1  
print(n1)  
print(n2)  
for i in range(1,n-1):  
 n3=n2+n1  
 n1=n2  
 n2=n3  
 print(n3)

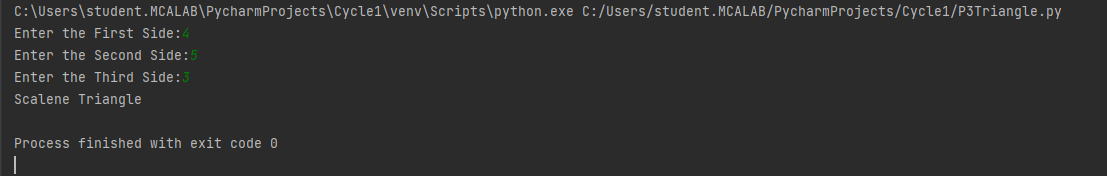
**Output:**



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

a = int(input("Enter the First Side:"))  
b = int(input("Enter the Second Side:"))  
c = int(input("Enter the Third Side:"))  
if a==b and b==c :  
 print("Equilateral Triangle")  
elif a==b or a==c or b==c :  
 print("Isosceles Triangle")  
else :  
 print("Scalene Triangle")

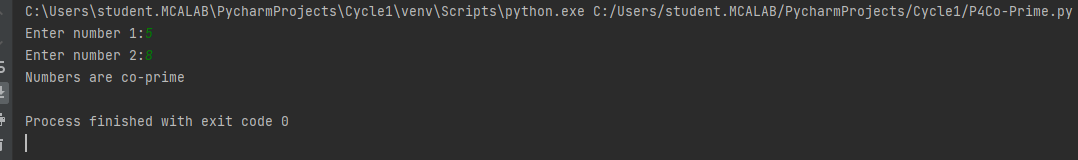
**Output:**



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

def gcd(a, b):  
 if b == 0:  
 return a  
 else:  
 return gcd(b, a % b)  
  
  
n1 = int(input("Enter number 1:"))  
n2 = int(input("Enter number 2:"))  
GCD = gcd(n1, n2)  
if GCD == 1:  
 print("Numbers are co-prime")  
else:

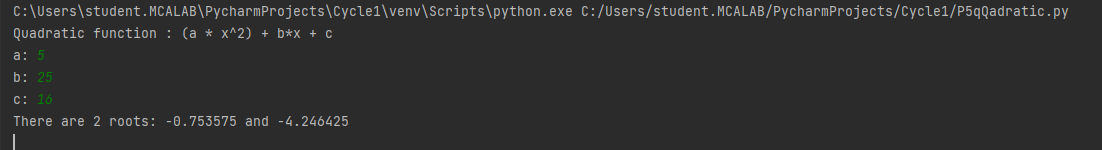
**Output:**



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

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()

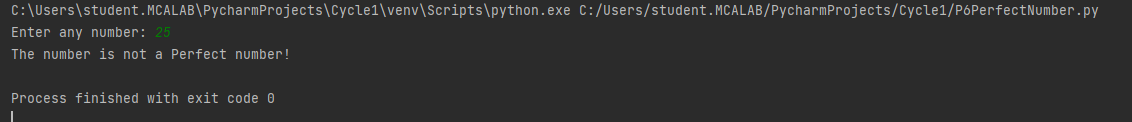
**Output:**



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

n = int(input("Enter any number: "))  
sum = 0  
for i in range(1, n):  
 if(n % i == 0):  
 sum = sum + i  
if (sum == n):  
 print("The number is a Perfect number!")  
else:  
 print("The number is not a Perfect number!")

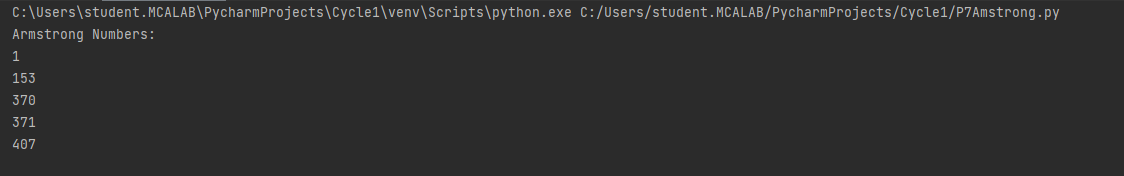
**Output:**



1. Program to display amstrong numbers upto 1000.

print("Armstrong Numbers:")  
for n in range(1,1000):  
 temp=n  
 sum=0  
 while temp>0:  
 a=temp%10  
 sum=sum+a\*\*3  
 temp=temp//10  
 if sum==n:  
 print (n)

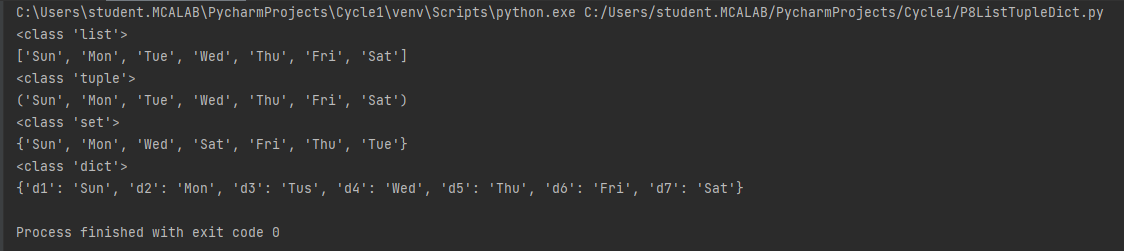
**Output:**



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.

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" : "Tus",  
 "d4" : "Wed",  
 "d5" : "Thu",  
 "d6" : "Fri",  
 "d7" : "Sat"  
}  
print(type(dict))  
print(dict)

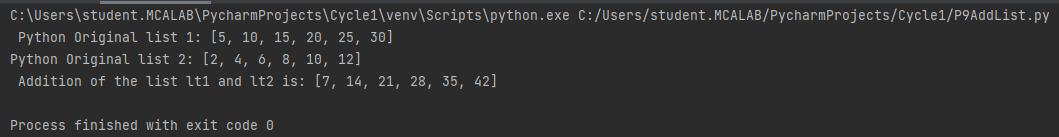
**Output:**



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

l1 = [5, 10, 15, 20, 25, 30]  
l2 = [2, 4, 6, 8, 10, 12]  
print ( " Python Original list 1: " + str (l1))  
print ( "Python Original list 2: " + str (l2))  
res = []  
for x in range (0, len (l1)):  
 res.append( l1[x] + l2[x])  
print ( " Addition of the list lt1 and lt2 is: " + str (res))

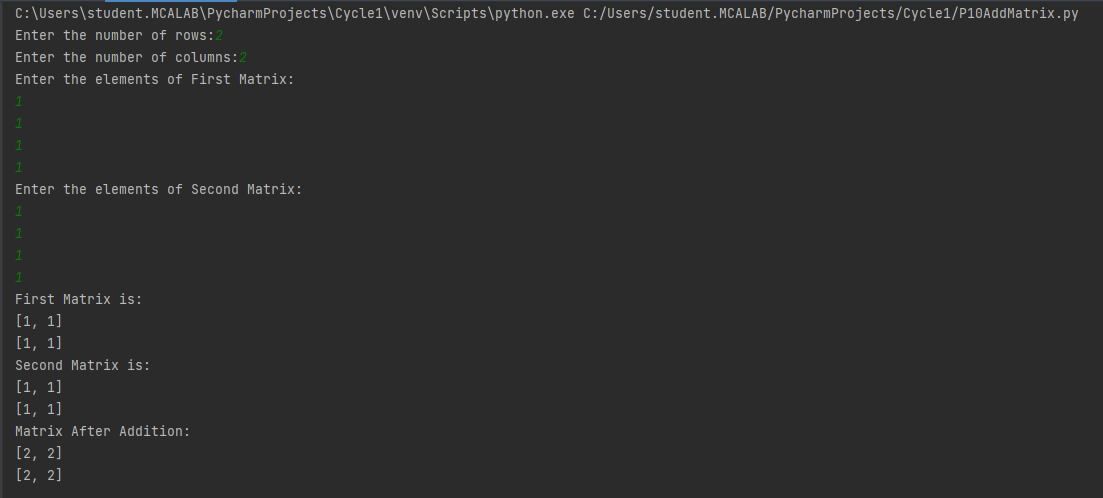
**Output:**



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

r=int(input("Enter the number of rows:"))  
c=int(input("Enter the number of columns:"))  
  
print("Enter the elements of First Matrix:")  
m1= [[int(input()) for i in range(c)] for i in range(r)]  
  
print("Enter the elements of Second Matrix:")  
m2=[[int(input()) for i in range(c)] for i in range(r)]  
  
print("First Matrix is: ")  
for n in m1:  
 print(n)  
  
  
print("Second Matrix is:")  
for n in m2:  
 print(n)  
  
r=[[0 for i in range(c)] for i in range(r)]  
  
print("Matrix After Addition:")  
for i in range(len(m1)):  
 for j in range(len(m2)):  
 r[i][j] = m1[i][j] + m2[i][j]  
  
for r1 in r:  
 print(r1)

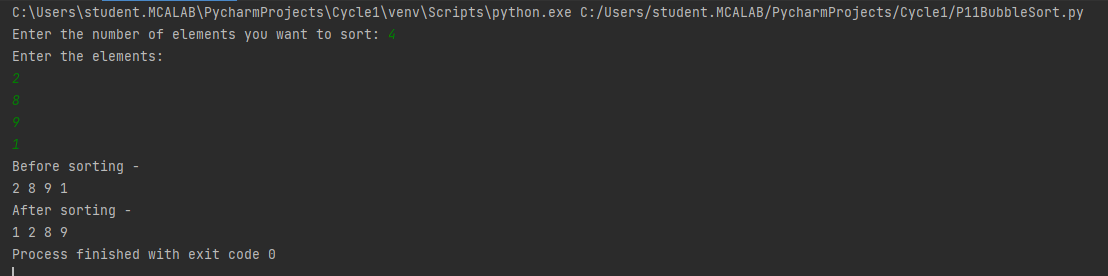
**Output:**



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

n = int(input("Enter the number of elements you want to sort: "))  
a = []  
print("Enter the elements: ")  
for i in range(n):  
 a.append(int(input()))  
print("Before sorting - ")  
for i in a:  
 print(i, end=" ")  
for i in range(0, len(a)):  
 for j in range(i + 1, len(a)):  
 if a[j] < a[i]:  
 temp = a[j]  
 a[j] = a[i]  
 a[i] = temp  
print("\nAfter sorting - ")  
for i in a:  
 print(i, end=" ")

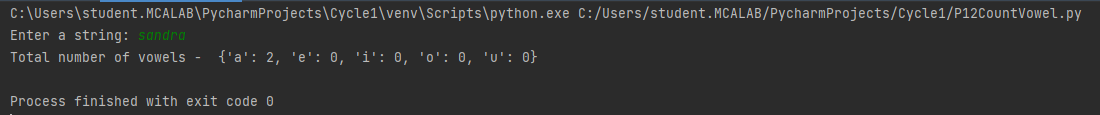
**Output:**



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

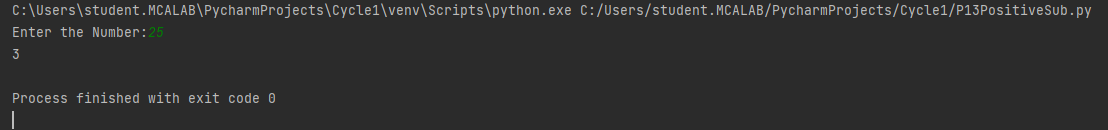
vowels = 'aeiou'  
user\_input = input("Enter a string: ")  
string = user\_input.casefold()  
count\_vowels = {}.fromkeys(vowels, 0)  
for x in string:  
 if x in count\_vowels:  
 count\_vowels[x] += 1  
  
print("Total number of vowels - ", count\_vowels)

**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.

**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.

phNo = input("Enter the phone 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(phNo))

**Output:**

