**Data Science and Machine Learning**

**Lab Cycle 1**

**Rinu Sebastian**

**20MCA043**

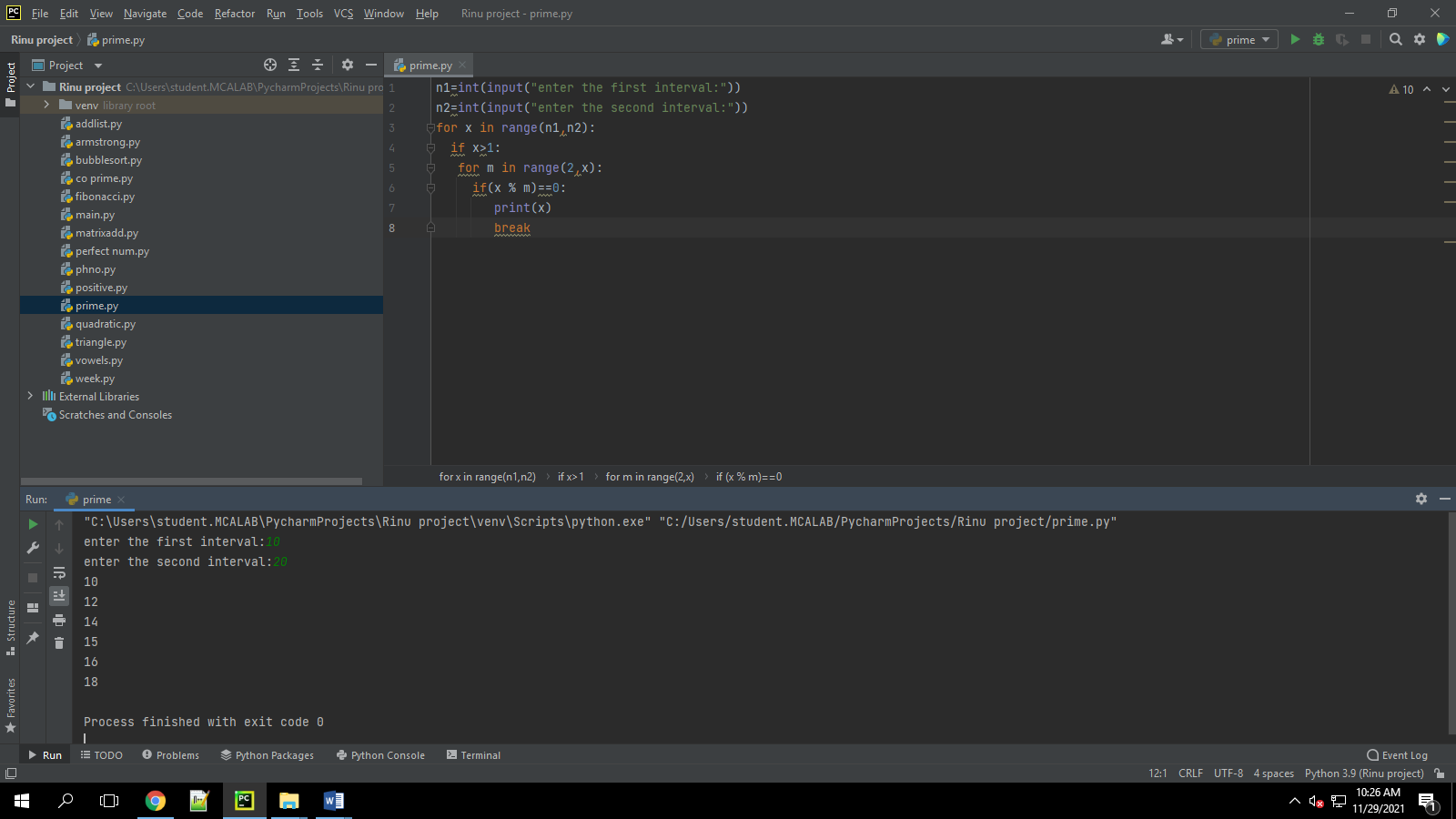
**S3MCA**

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

**Program**

n1=int(input("enter the first interval:"))  
n2=int(input("enter the second interval:"))  
for x in range(n1,n2):  
 if x>1:  
 for m in range(2,x):  
 if(x % m)==0:  
 print(x)  
 break

**Output**

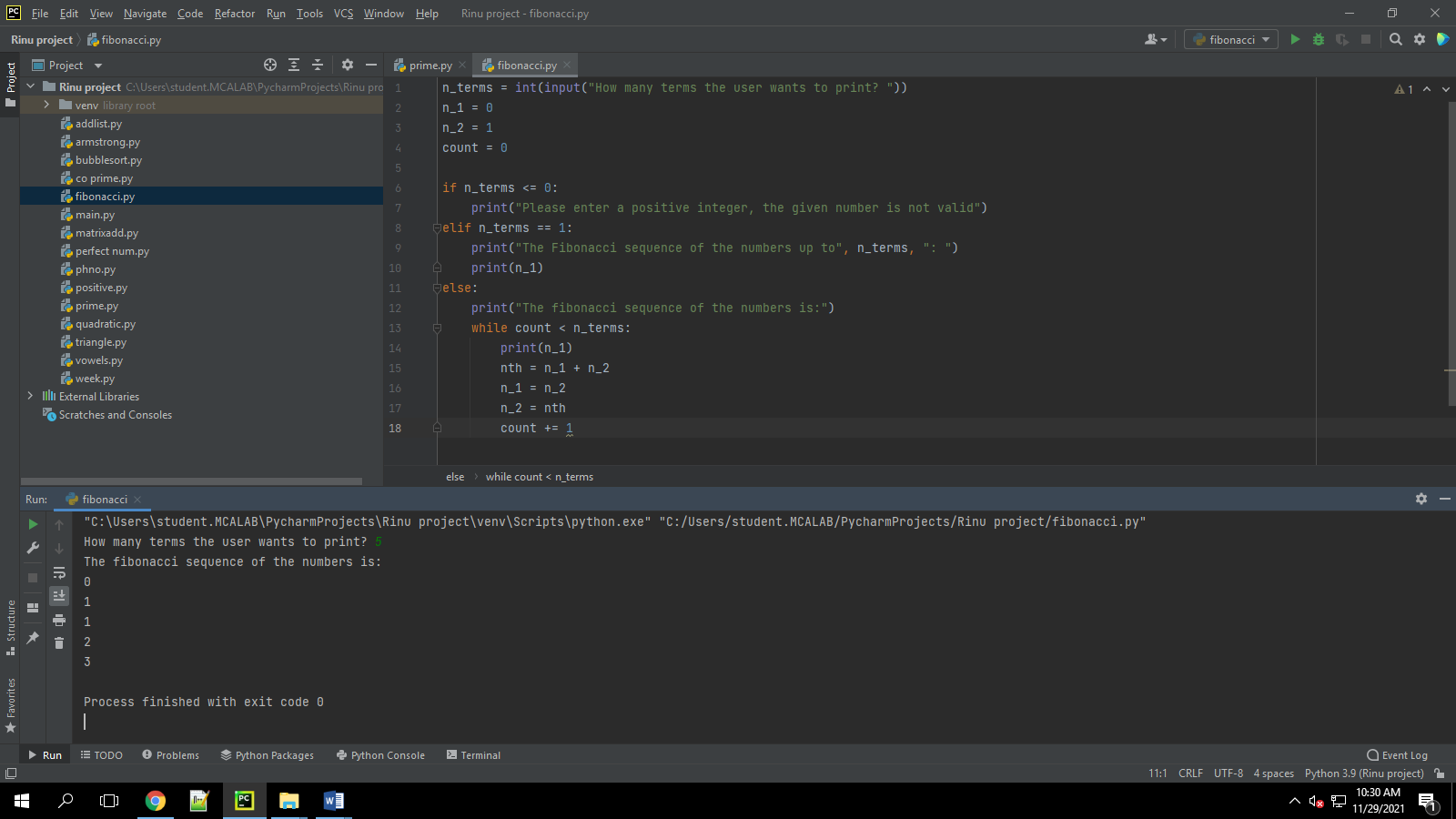


1. Program to print the first N Fibonacci numbers.

**Program**

n\_terms = int(input("How many terms the user wants to print? "))  
n\_1 = 0  
n\_2 = 1  
count = 0  
  
if n\_terms <= 0:  
 print("Please enter a positive integer, the given number is not valid")  
elif n\_terms == 1:  
 print("The Fibonacci sequence of the numbers up to", n\_terms, ": ")  
 print(n\_1)  
else:  
 print("The fibonacci sequence of the numbers is:")  
 while count < n\_terms:  
 print(n\_1)  
 nth = n\_1 + n\_2  
 n\_1 = n\_2  
 n\_2 = nth  
 count += 1

**Output**

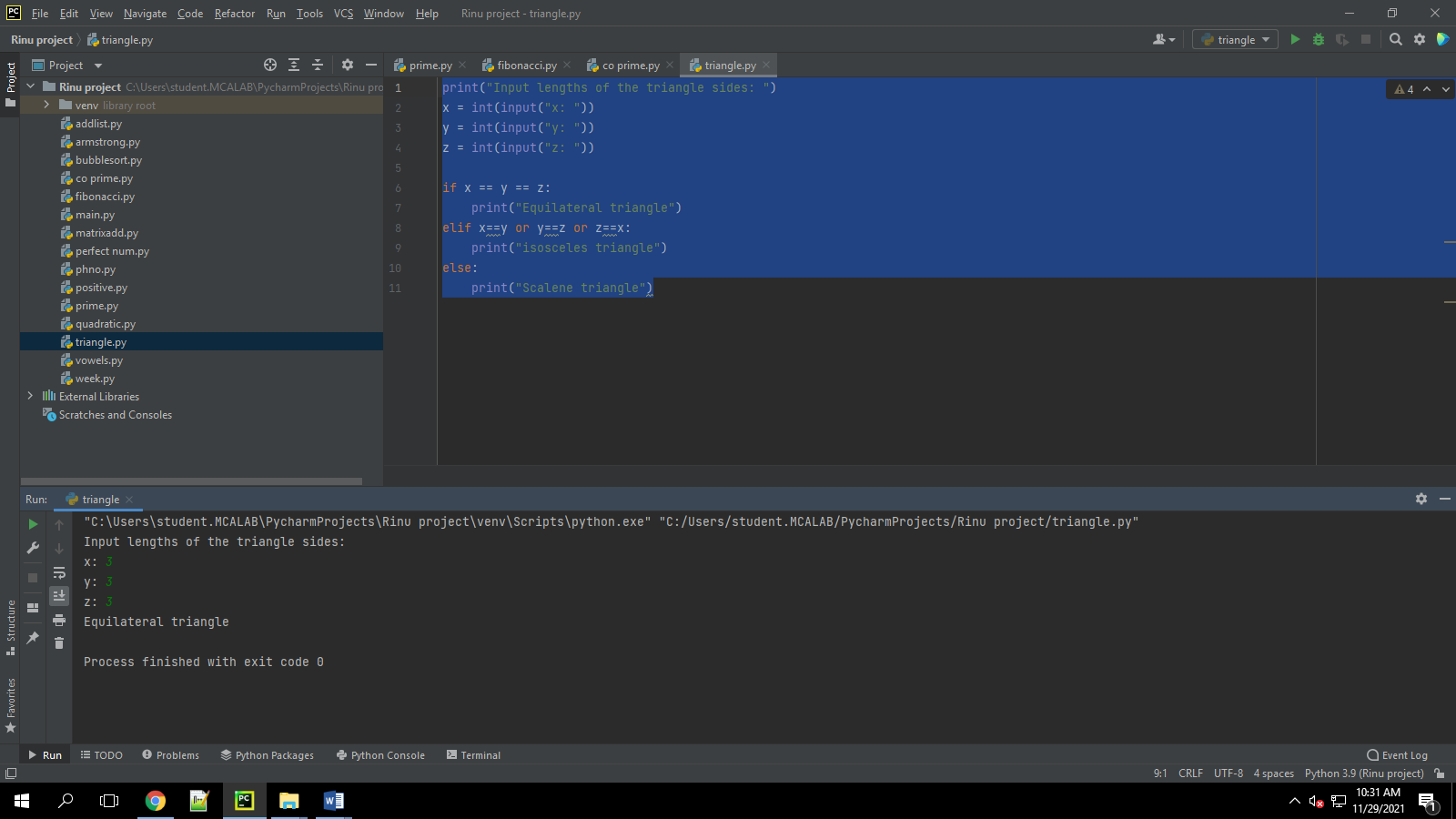


1. Given sides of a triangle, write a program to check whether given triangle is an isosceles, equilateral or scalene.

**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**

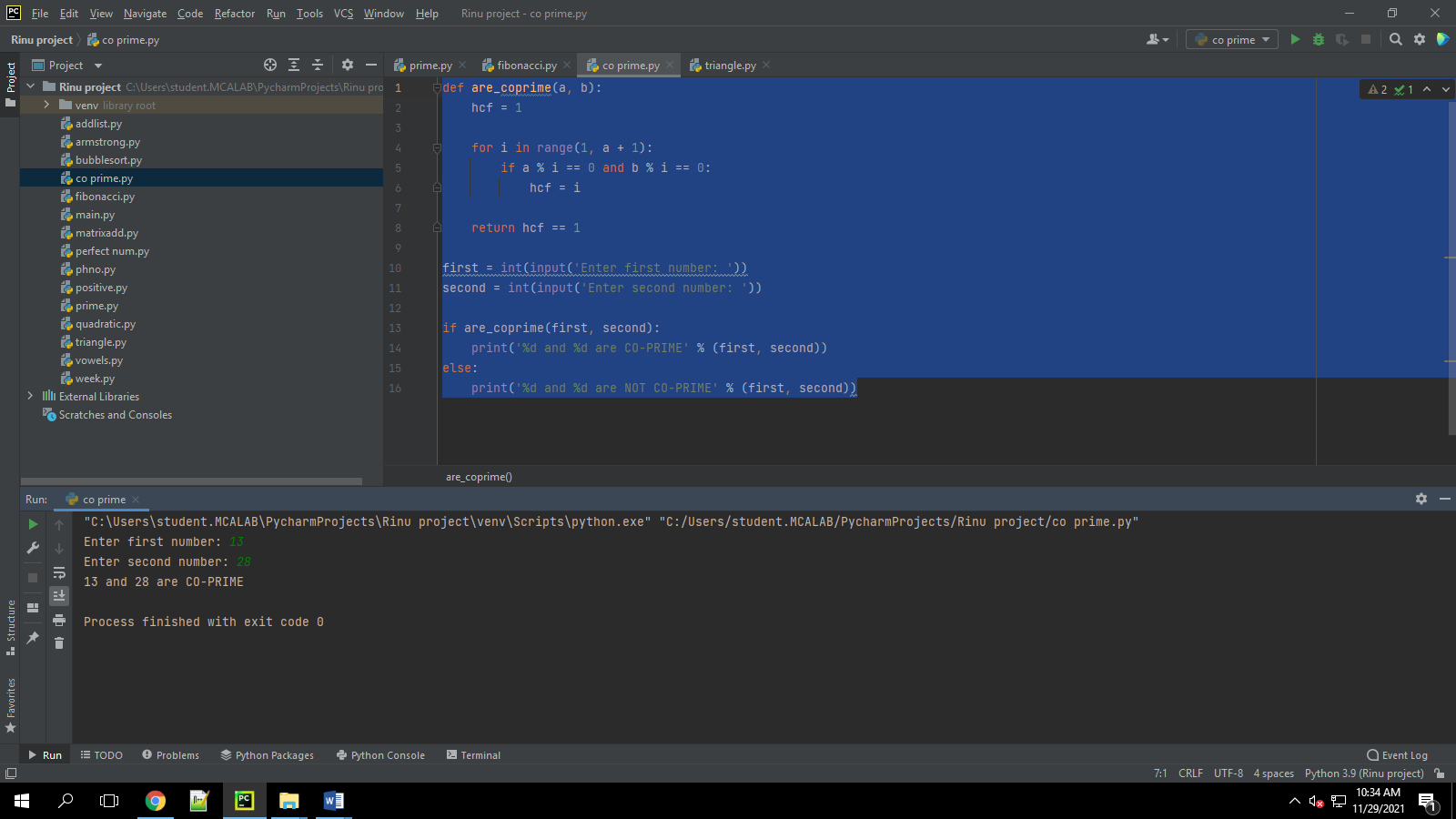


1. Program to check whether given pair of number is coprime

**Program**

def 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  
  
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))

**Output**

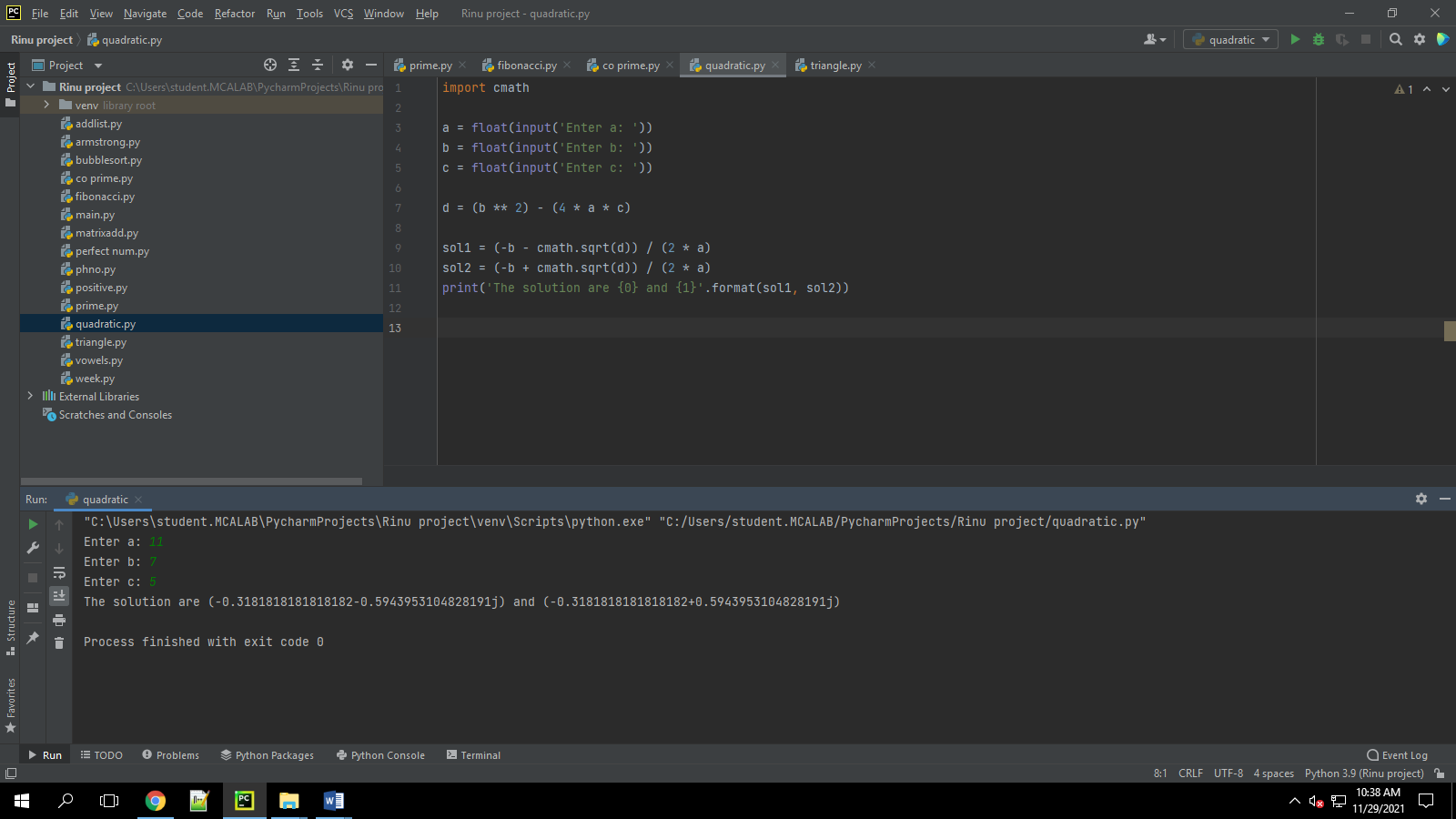


1. Program to find the roots of a quadratic equation(rounded to 2 decimal places)

**Program**

import cmath  
  
a = float(input('Enter a: '))  
b = float(input('Enter b: '))  
c = float(input('Enter c: '))  
  
d = (b \*\* 2) - (4 \* a \* c)  
  
sol1 = (-b - cmath.sqrt(d)) / (2 \* a)  
sol2 = (-b + cmath.sqrt(d)) / (2 \* a)  
print('The solution are {0} and {1}'.format(sol1, sol2))

**Output**

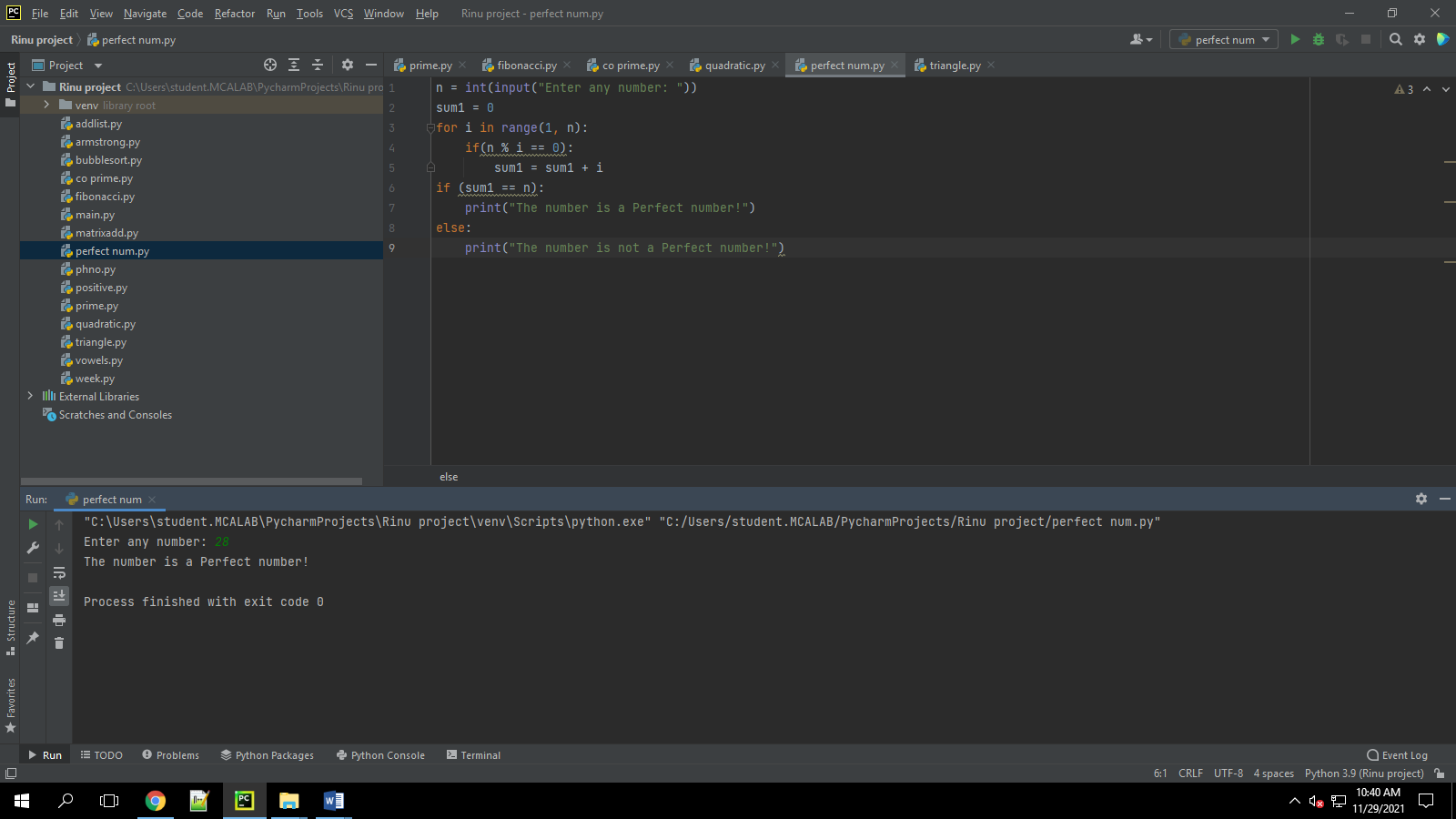


1. Program to check whether a given number is perfect number or not(sum of factors =number)

**Program**

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!")

**Output**

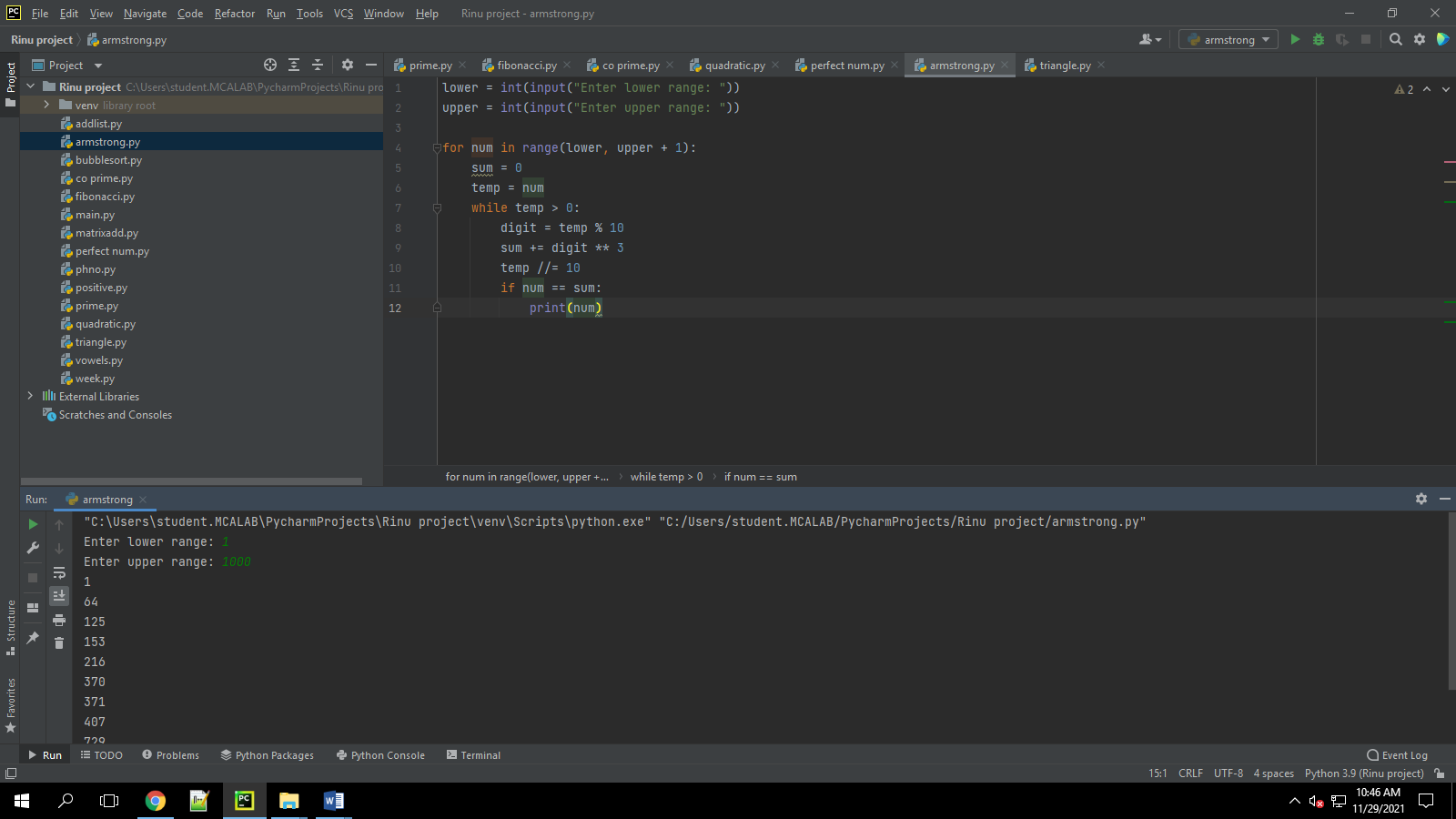


1. Program to display amstrong numbers upto 1000

**Program**

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**

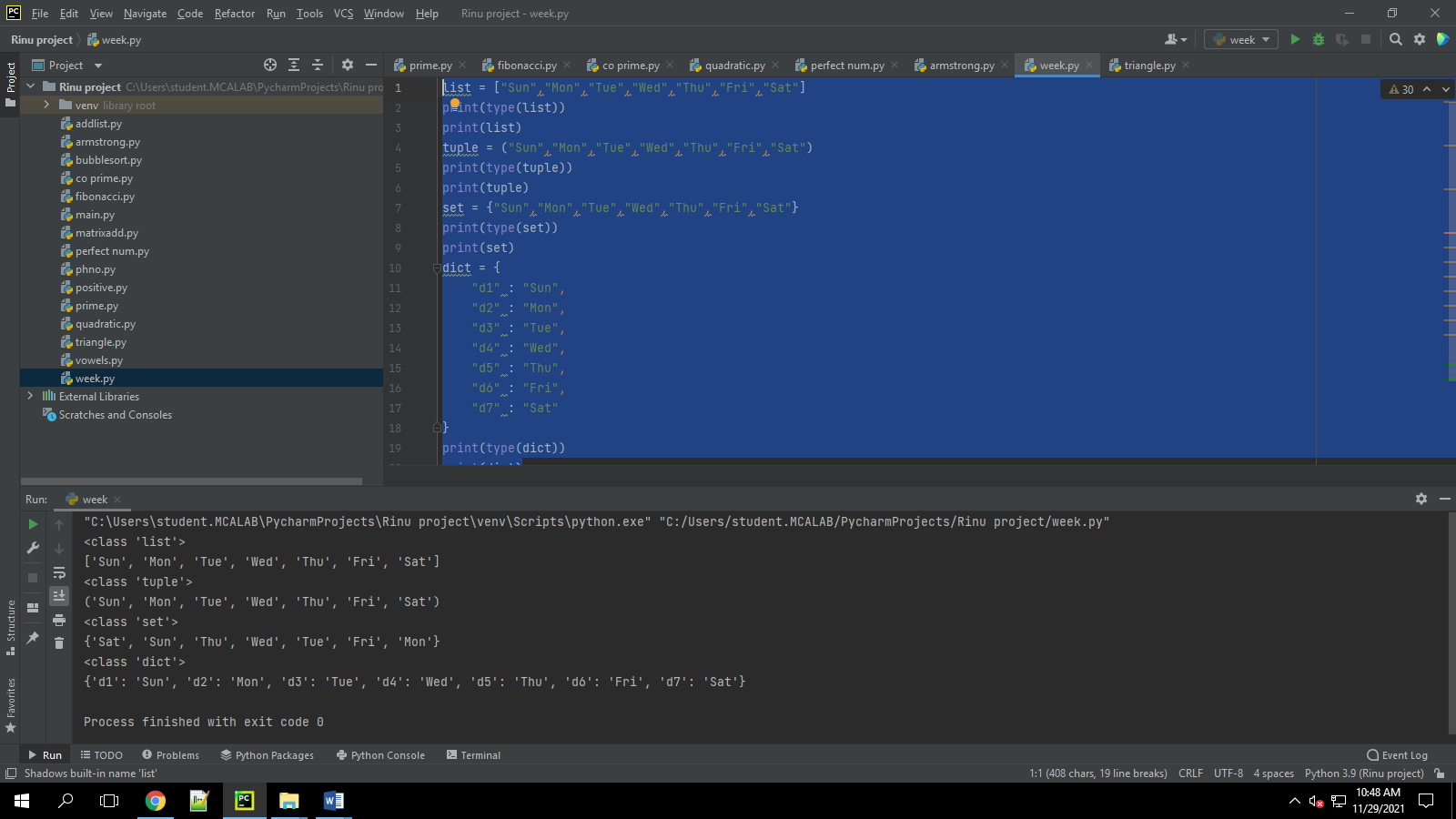


1. Store and display the days of a week as a **List, Tuple, Dictionary, Set.** Also demonstrate different ways to store values in each of them. Display its type also.

**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**

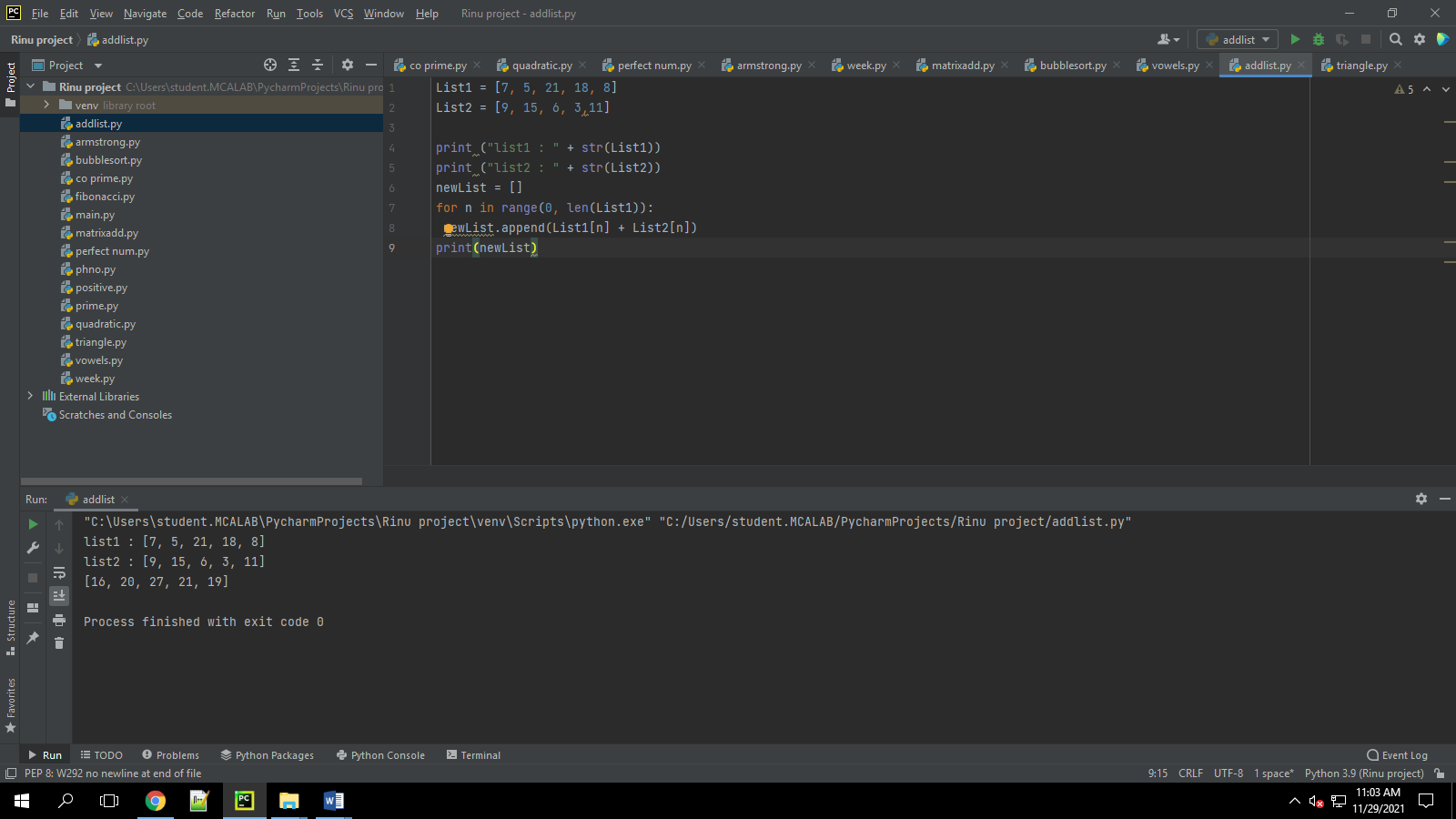


1. Write a program to add elements of given 2 lists

**Program**

List1 = [7, 5, 21, 18, 8]  
List2 = [9, 15, 6, 3,11]  
  
print ("list1 : " + str(List1))  
print ("list2 : " + str(List2))  
newList = []  
for n in range(0, len(List1)):  
 newList.append(List1[n] + List2[n])  
print(newList)

**Output**

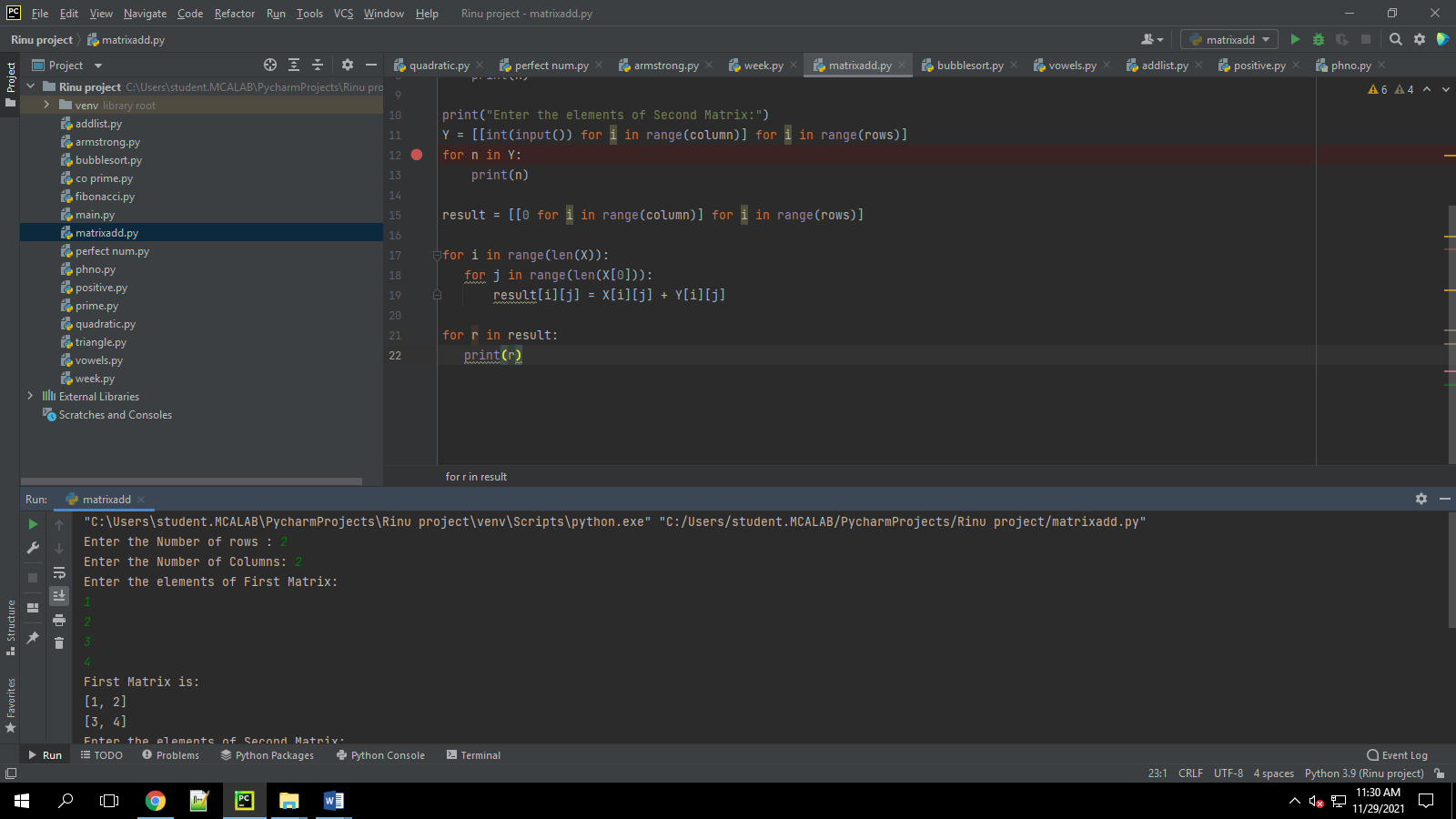


1. Write a program to find the sum of 2 matrices using nested List.

**Program**

rows = int(input("Enter the Number of rows : "))  
column = int(input("Enter the Number of Columns: "))  
  
print("Enter the elements of First Matrix:")  
X = [[int(input()) for i in range(column)] for i in range(rows)]  
print("First Matrix is: ")  
for n in X:  
 print(n)  
  
print("Enter the elements of Second Matrix:")  
Y = [[int(input()) for i in range(column)] for i in range(rows)]  
for n in Y:  
 print(n)  
  
result = [[0 for i in range(column)] for i in range(rows)]  
  
for i in range(len(X)):  
 for j in range(len(X[0])):  
 result[i][j] = X[i][j] + Y[i][j]  
  
for r in result:  
 print(r)

**Output**

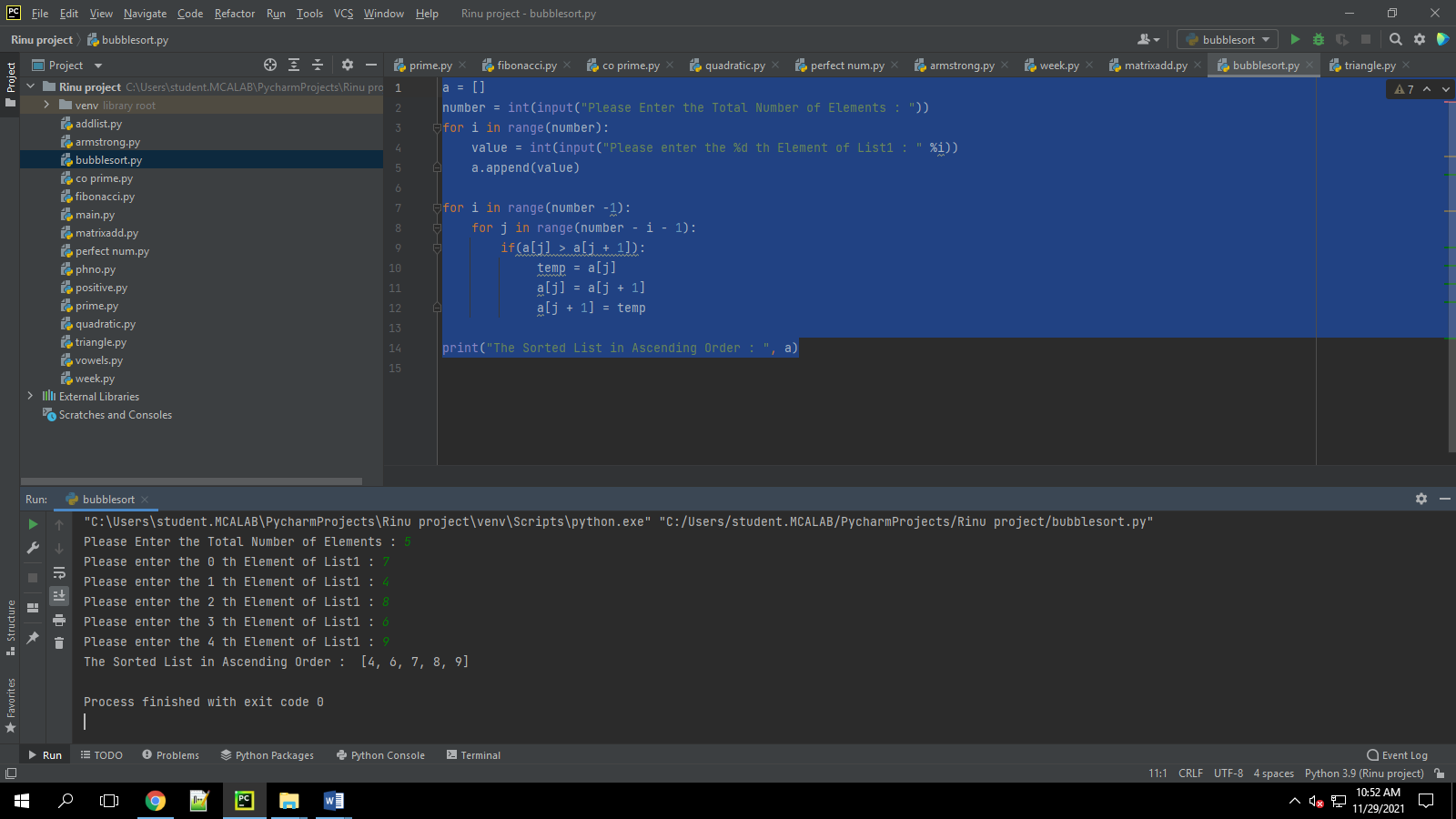


1. Write a program to perform bubble sort on a given set of elements.

**Program**

a = []  
number = int(input("Please Enter the Total Number of Elements : "))  
for i in range(number):  
 value = int(input("Please enter the %d th Element of List1 : " %i))  
 a.append(value)  
  
for i in range(number -1):  
 for j in range(number - i - 1):  
 if(a[j] > a[j + 1]):  
 temp = a[j]  
 a[j] = a[j + 1]  
 a[j + 1] = temp  
  
print("The Sorted List in Ascending Order : ", a)

**Output**

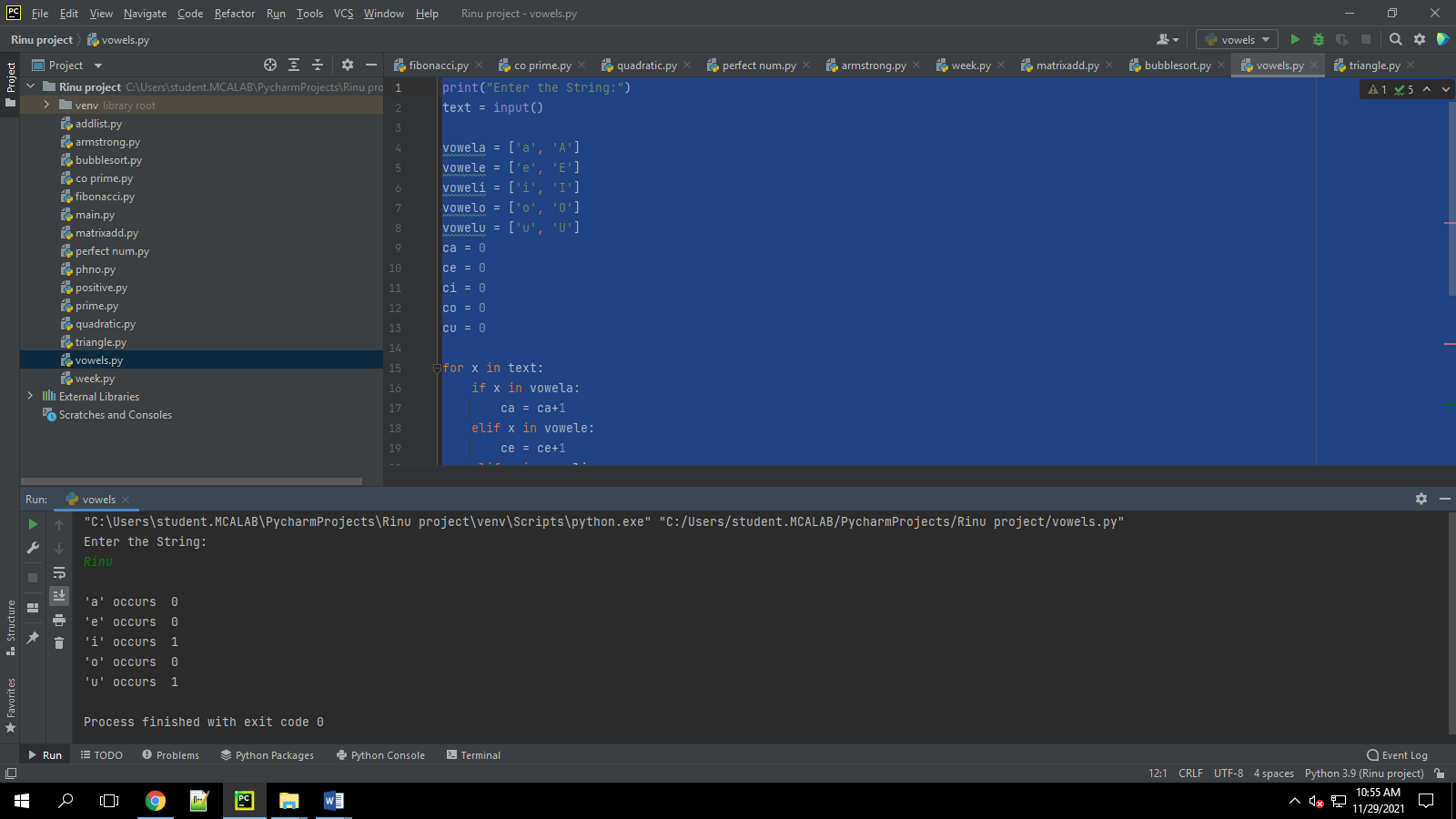


1. Program to find the count of each vowel in a string(use dictionary)

**Program**

print("Enter the String:")  
text = input()  
  
vowela = ['a', 'A']  
vowele = ['e', 'E']  
voweli = ['i', 'I']  
vowelo = ['o', 'O']  
vowelu = ['u', 'U']  
ca = 0  
ce = 0  
ci = 0  
co = 0  
cu = 0  
  
for x in text:  
 if x in vowela:  
 ca = ca+1  
 elif x in vowele:  
 ce = ce+1  
 elif x in voweli:  
 ci = ci+1  
 elif x in vowelo:  
 co = co+1  
 elif x in vowelu:  
 cu = cu+1  
  
print("\n'a' occurs ", ca)  
print("'e' occurs ", ce)  
print("'i' occurs ", ci)  
print("'o' occurs ", co)  
print("'u' occurs ", cu)

**Output**

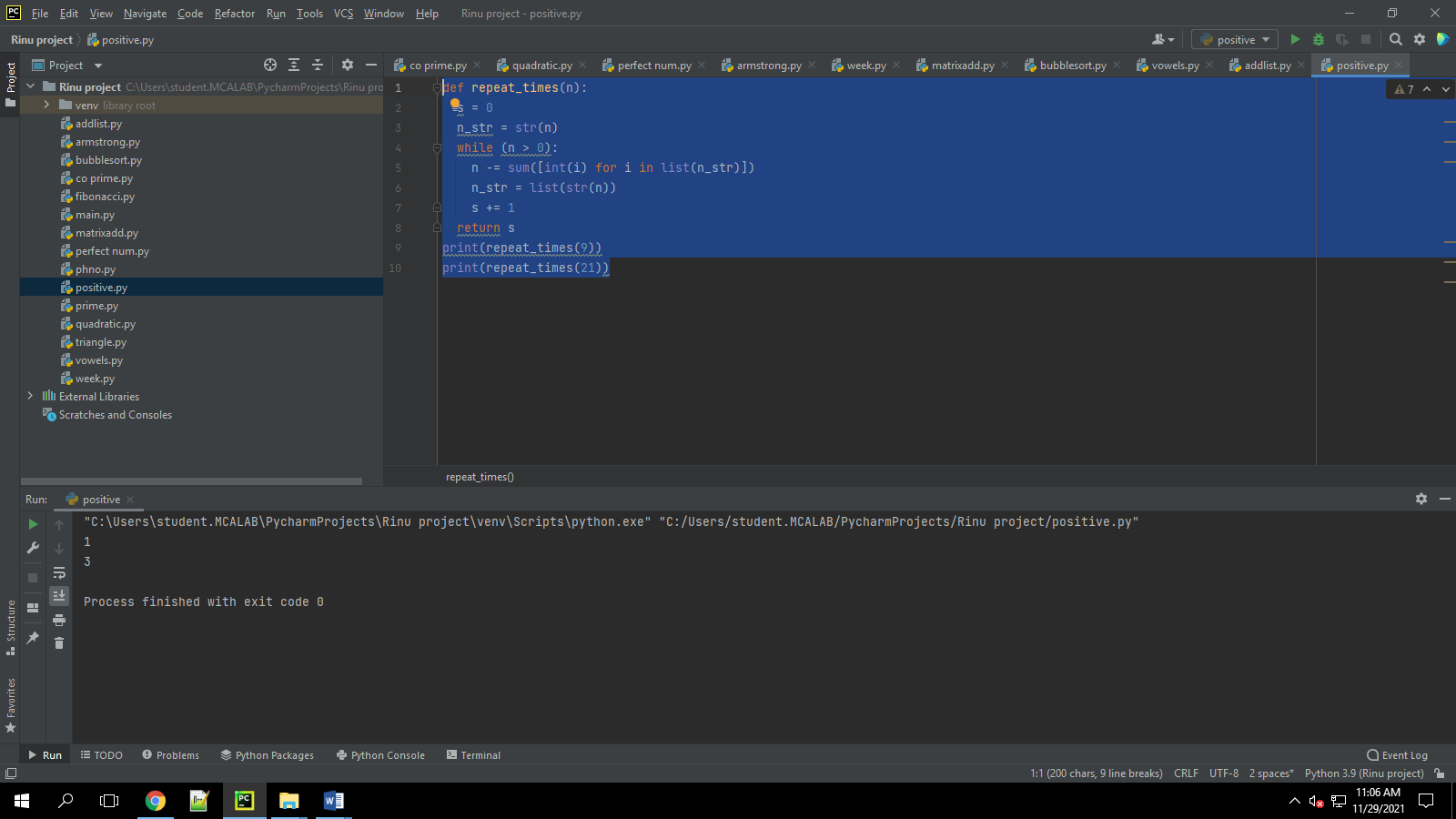


1. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is positive

**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  
print(repeat\_times(9))  
print(repeat\_times(21))

**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**

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([9,8,3,2,2,0,9,7,6,3]))

**Output**

