DATA SCIENCE AND MACHINE LEARNING:

**LAB CYCLE 1**

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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def is\_prime(num):

if num <= 1:

return False

for i in range (2,num):

if num % i == 0:

return False

return True

start=int(input("Enter the starting number: "))

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

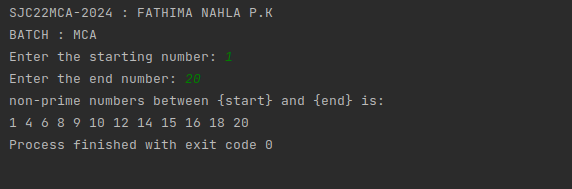
print("non-prime numbers between {start} and {end} is: ")

for num in range(start,end+1):

if not is\_prime(num):

print(num,end=" ")

OUTPUT:



2. Program to print the first N Fibonacci numbers.

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

n = int(input("how many terms to print : "))

num1 = 0

num2 = 1

count = 0

if n <= 0:

print("enter positive integer")

elif n == 1:

print("sequence of the numbers upto",n,":")

print(num1)

else:

print("sequence is")

while count < n:

print(num1)

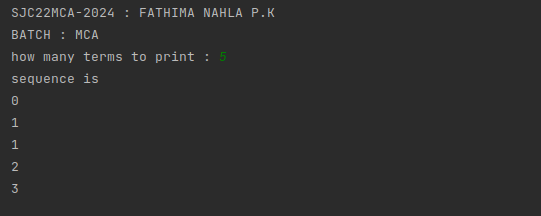
num3 = num1 + num2

num1 = num2

num2 = num3

count += 1

OUTPUT:



3. Given sides of a triangle, write a program to check whether given triangle is an

isosceles, equilateral or scalene.

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def checktraingle(x, y, z):

if x == y == z:

print (" equilateral triangle")

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

print (" isosceles triangle")

else:

print("scalene triangle")

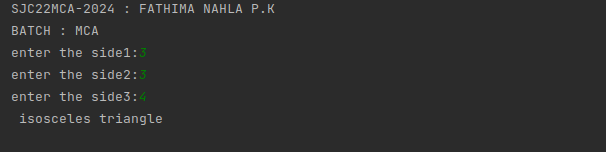
x = int(input("enter the side1:"))

y = int(input("enter the side2:"))

z = int(input("enter the side3:"))

checktraingle(x, y, z)

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def 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 the first number :"))

second = int(input("enter the second number :"))

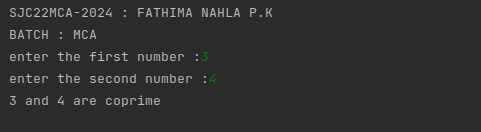
if coprime(first, second):

print("%d and %d are coprime" %(first, second))

else:

print("%d and %d are not coprime" % (first, second))

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

a=int(input("enter a :"))

b=int(input("enter b :"))

c=int(input("enter c :"))

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

d1=d\*\*0.5

if(d<0):

print("roots are imaginory")

else:

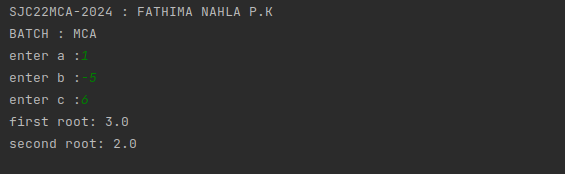
r1=(-b+d1)/2\*a

r2=(-b-d1)/2\*a

print("first root:",round(r1,2))

print("second root:",round(r2,2))

OUTPUT:



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

=number)

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

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

sum=0

for i in range(1,num):

if(num%i==0):

sum=sum+i

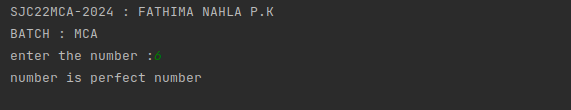
if(sum==num):

print("number is perfect number")

else:

print("number is not a perfect number")

OUTPUT:



7. Program to display amstrong numbers upto 1000

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

t1=1

t2=1000

for num in range(t1, t2+1):

order = len(str(num))

sum = 0

temp = num

while temp>0:

digit = temp % 10

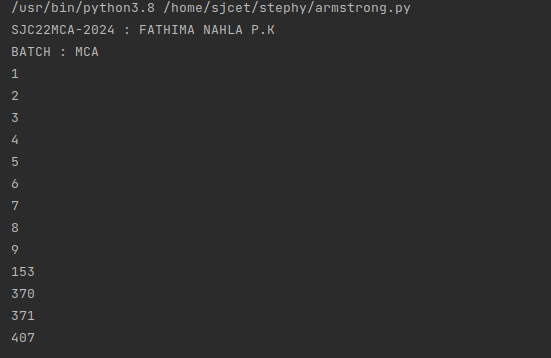
sum += digit \*\* order

temp //= 10

if num == sum:

print(num)

OUTPUT:



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:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

days\_list = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

print("List:", days\_list)

print("Type of List:", type(days\_list))

days\_tuple = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")

print("Tuple:", days\_tuple)

print("Type of Tuple:", type(days\_tuple))

days\_dict = {1: "Monday", 2: "Tuesday", 3: "Wednesday", 4: "Thursday", 5: "Friday", 6: "Saturday", 7: "Sunday"}

print("Dictionary:", days\_dict)

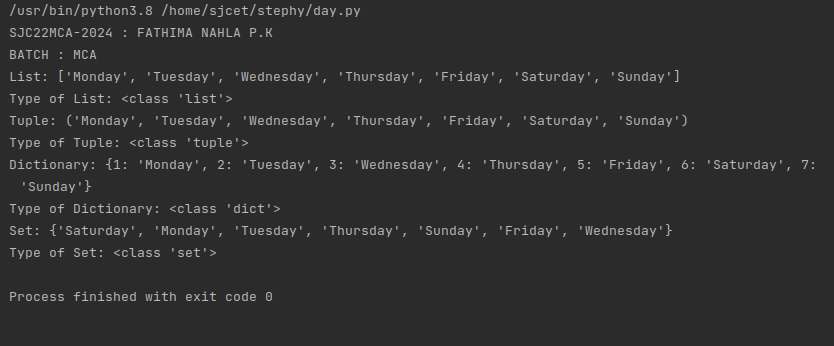
print("Type of Dictionary:", type(days\_dict))

days\_set = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"}

print("Set:", days\_set)

print("Type of Set:", type(days\_set))

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

list1=[1,2,3,4,5]

list2=[6,7,8,9,10]

print("original list1 : " + str(list1))

print("original list2 : " + str(list2))

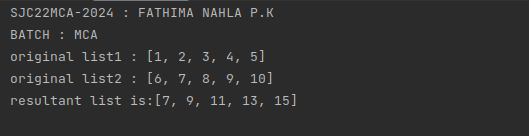
list3=[]

for i in range(0,len(list1)):

list3.append(list1[i] + list2[i])

print ("resultant list is:" + str(list3))

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

n=int(input("enter the order :"))

m1=[]

print("enter the elements :")

for i in range(n):

m1.append([])

for j in range(n):

num=int(input())

m1[i].append(num)

m2=[]

print("enter the elements :")

for i in range(n):

m2.append([])

for j in range(n):

num=int(input())

m2[i].append(num)

print("first matrix is :")

for i in range(n):

for j in range(n):

print(m1[i][j],end=" ")

print()

print("second matrix is :")

for i in range(n):

for j in range(n):

print(m2[i][j],end=" ")

print()

m3=[]

for i in range(n):

m3.append([])

for j in range(n):

m3[i].append(m1[i][j]+m2[i][j])

print("resultant matrix is :")

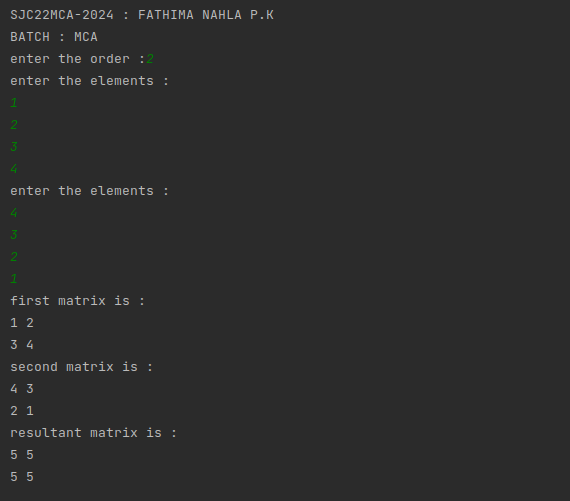
for i in range(n):

for j in range(n):

print(m3[i][j],end=" ")

print()

OUTPUT:



CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def input\_matrix(rows, cols):

matrix = []

for i in range(rows):

row = []

for j in range(cols):

element = int(input("Enter the elements: "))

row.append(element)

matrix.append(row)

return matrix

def add\_matrices(matrix1, matrix2):

if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]):

return None

result = []

for i in range(len(matrix1)):

row = []

for j in range(len(matrix1[0])):

element = matrix1[i][j] + matrix2[i][j]

row.append(element)

result.append(row)

return result

def display\_matrix(matrix):

for row in matrix:

for element in row:

print(element, end=" ")

print()

rows = int(input("Enter the number of rows: "))

cols = int(input("Enter the number of columns: "))

print("Enter elements of the first matrix:")

matrix1 = input\_matrix(rows, cols)

print("Enter elements of the second matrix:")

matrix2 = input\_matrix(rows, cols)

result\_matrix = add\_matrices(matrix1, matrix2)

if result\_matrix:

print("\nMatrix 1:")

display\_matrix(matrix1)

print("\nMatrix 2:")

display\_matrix(matrix2)

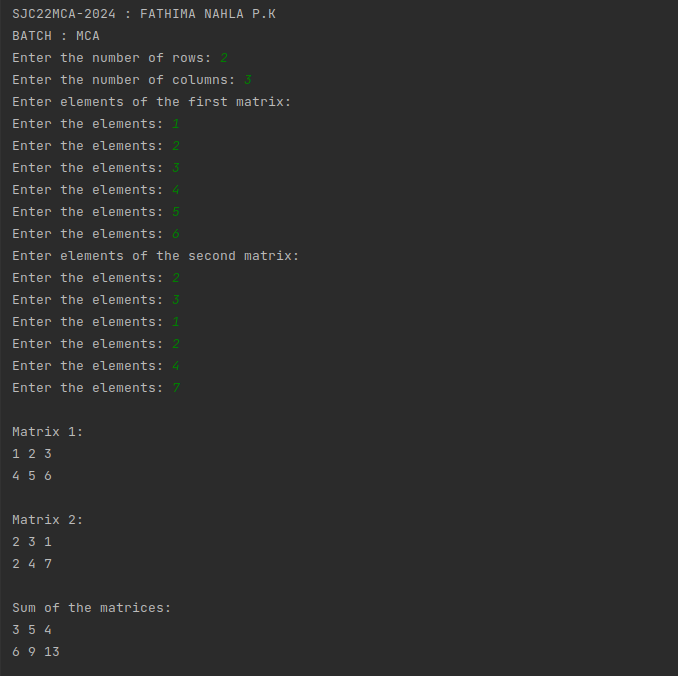
print("\nSum of the matrices:")

display\_matrix(result\_matrix)

else:

print("Matrix addition is not possible. Matrices are of different sizes.")

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def bubble\_sort(arr):

n = len(arr)

for i in range(n - 1):

swapped = False

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

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

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

swapped = True

if not swapped:

break

input\_str = input("Enter elements : ")

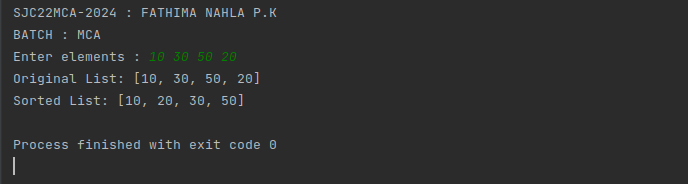
elements = [int(x) for x in input\_str.split()]

print("Original List:", elements)

bubble\_sort(elements)

print("Sorted List:", elements)

OUTPUT:



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

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def count\_vowels(input\_string):

vowel\_counts = {'a': 0, 'e': 0, 'i': 0, 'o': 0, 'u': 0}

input\_string = input\_string.lower()

for char in input\_string:

if char in vowel\_counts:

vowel\_counts[char] += 1

return vowel\_counts

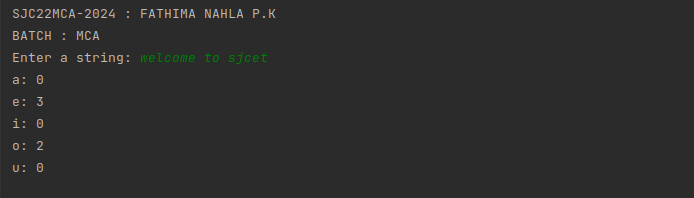
input\_string = input("Enter a string: ")

vowel\_count = count\_vowels(input\_string)

for vowel, count in vowel\_count.items():

print(f"{vowel}: {count}")

OUTPUT:



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(eg: 256-&gt;2+5+6=13

256-13=243

243-9=232……..

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def sum\_of\_digits(n):

digit\_sum = 0

while n > 0:

digit\_sum += n % 10

n //= 10

return digit\_sum

num = int(input("Enter a positive number: "))

while num > 0:

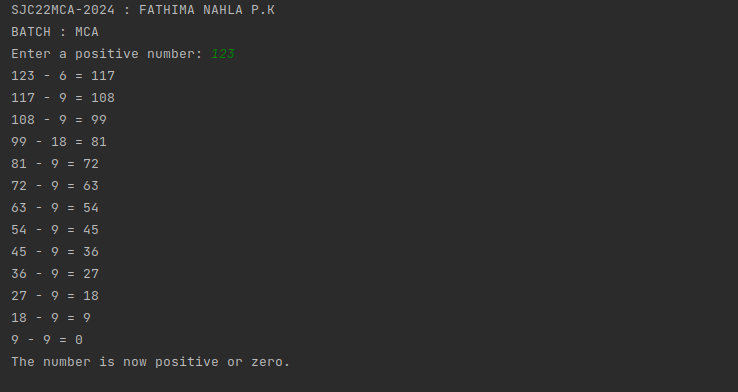
digit\_sum = sum\_of\_digits(num)

num -= digit\_sum

print(f"{num + digit\_sum} - {digit\_sum} = {num}")

print("The number is now positive or zero.")

OUTPUT:



14. Write a Python program that accepts a 10 digit mobile number, and find the digits

which are absent in a given mobile number

CODE:

print("SJC22MCA-2024 : FATHIMA NAHLA P.K")

print("BATCH : MCA")

def find\_absent\_digits(mobile\_number):

all\_digits = set("0123456789")

number\_digits = set(mobile\_number)

absent\_digits = all\_digits - number\_digits

return absent\_digits

mobile\_number = input("Enter a 10-digit mobile number: ")

if len(mobile\_number) == 10 and mobile\_number.isdigit():

absent\_digits = find\_absent\_digits(mobile\_number)

if absent\_digits:

print("Absent digits:", ', '.join(absent\_digits))

else:

print("All digits are present in the mobile number.")

else:

print("Invalid input. Please enter a 10-digit mobile number.")

OUTPUT:

