NAME:MANVITH BALAJI

SECTION:A

MIS NO. :112315115

1.

Input code:

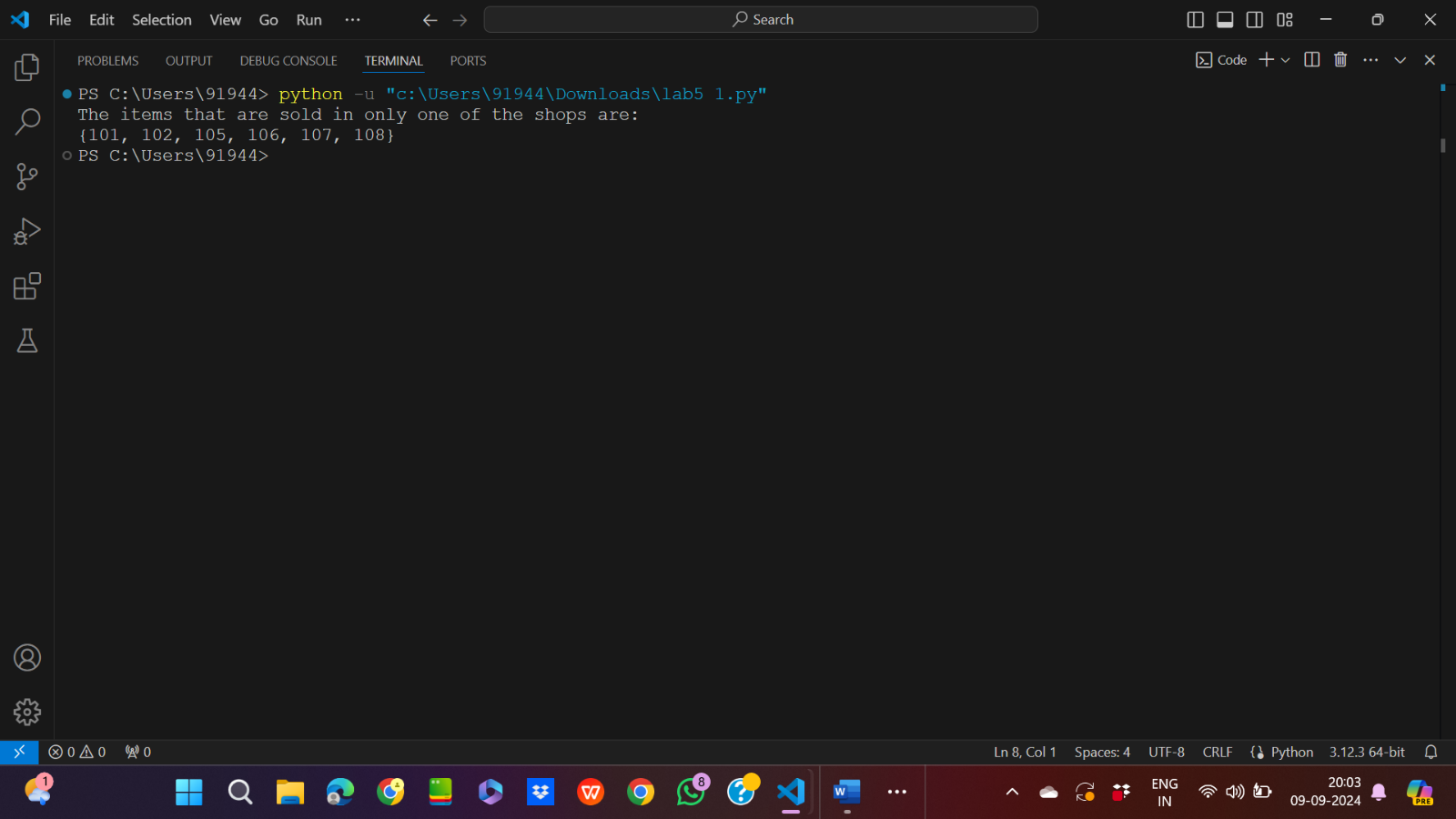
store1=[101,102,103,104,105]

store2=[103,104,106,107,108]

print("The items that are sold in only one of the shops are: ")

print(set(store1)^set(store2))

Output:



2.

Input code:

facebook\_friends={"Alice","Bob","Charlie"}

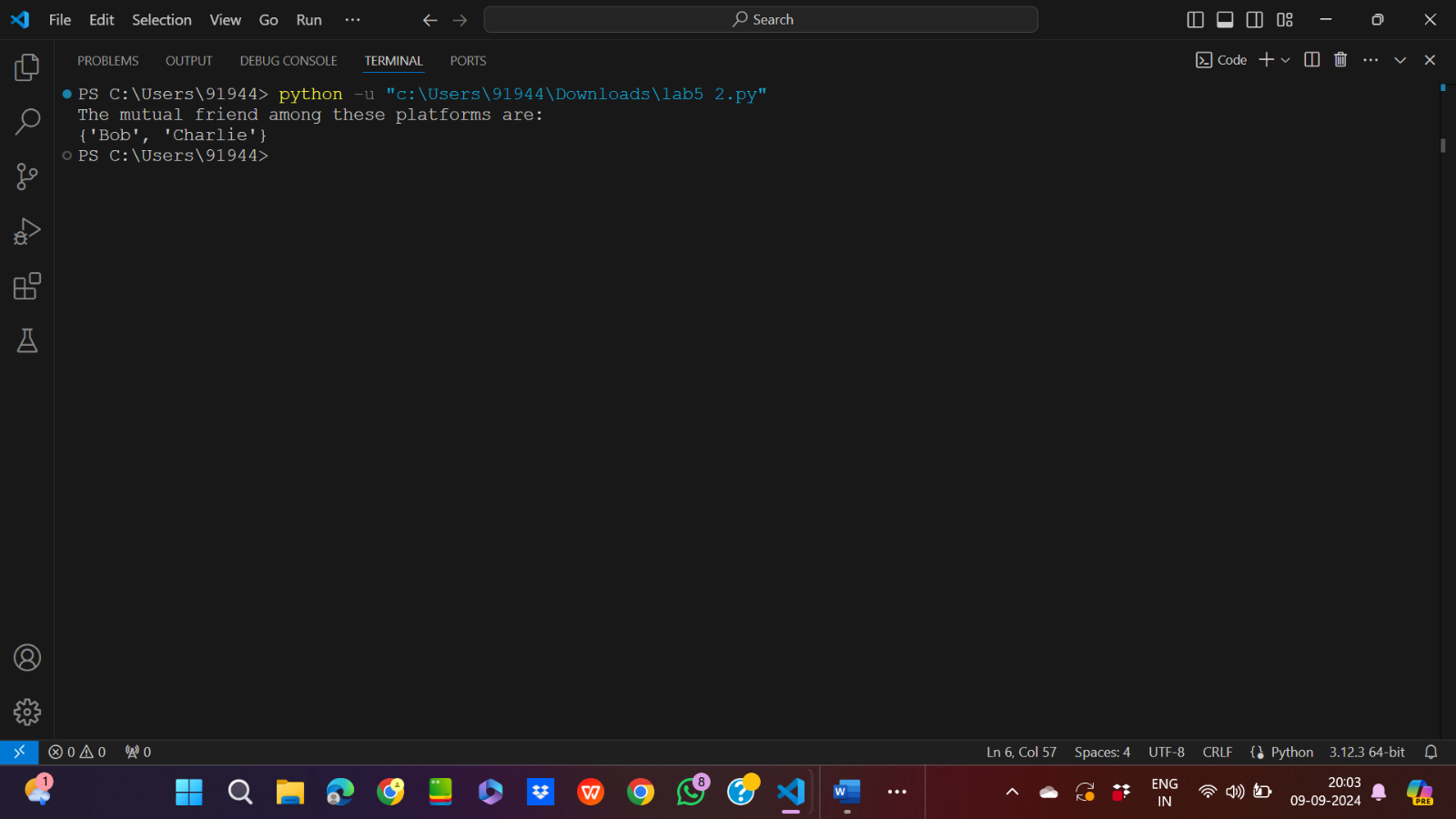
twitter\_friends={"Bob","Charlie","David"}

linkedin\_friends={"Bob","Charlie","Emma"}

print("The mutual friend among these platforms are: ")

print(facebook\_friends&twitter\_friends&linkedin\_friends)

Output:



3.

Input code:

a=input("Enter the list of integers with space: ")

num=a.split(" ")

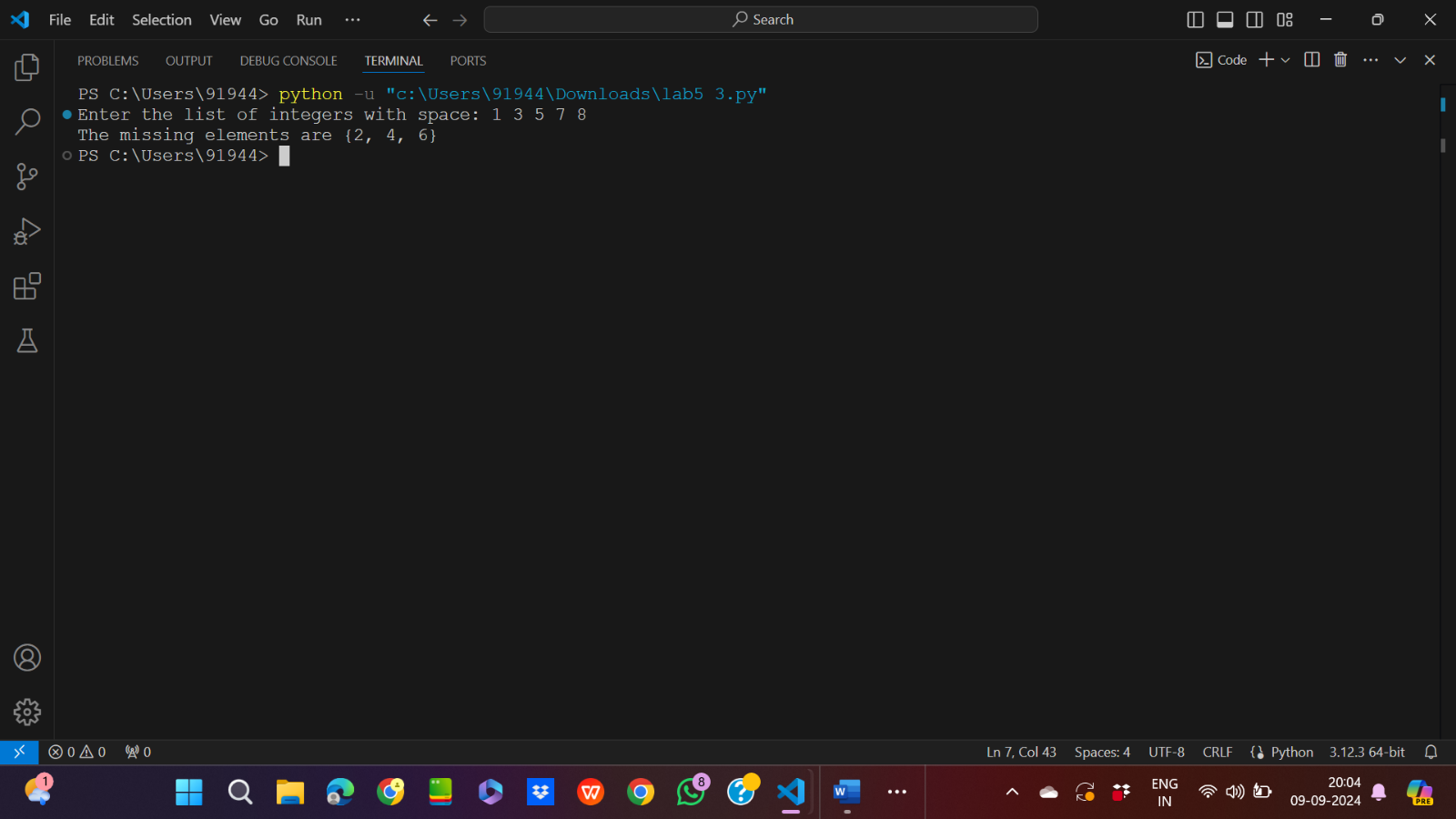
num=[int(j) for j in num]

num.sort(reverse=True)

myset={i for i in range(1,num[0]+1) if i not in num}

print(f"The missing elements are {myset}")

Output:



4.

Input code:

email\_address=["Apple.gmail.com","Meta.gmail.com","Facebook.gmail.com","Facebook.gmail.com","Apple.gmail.com"]

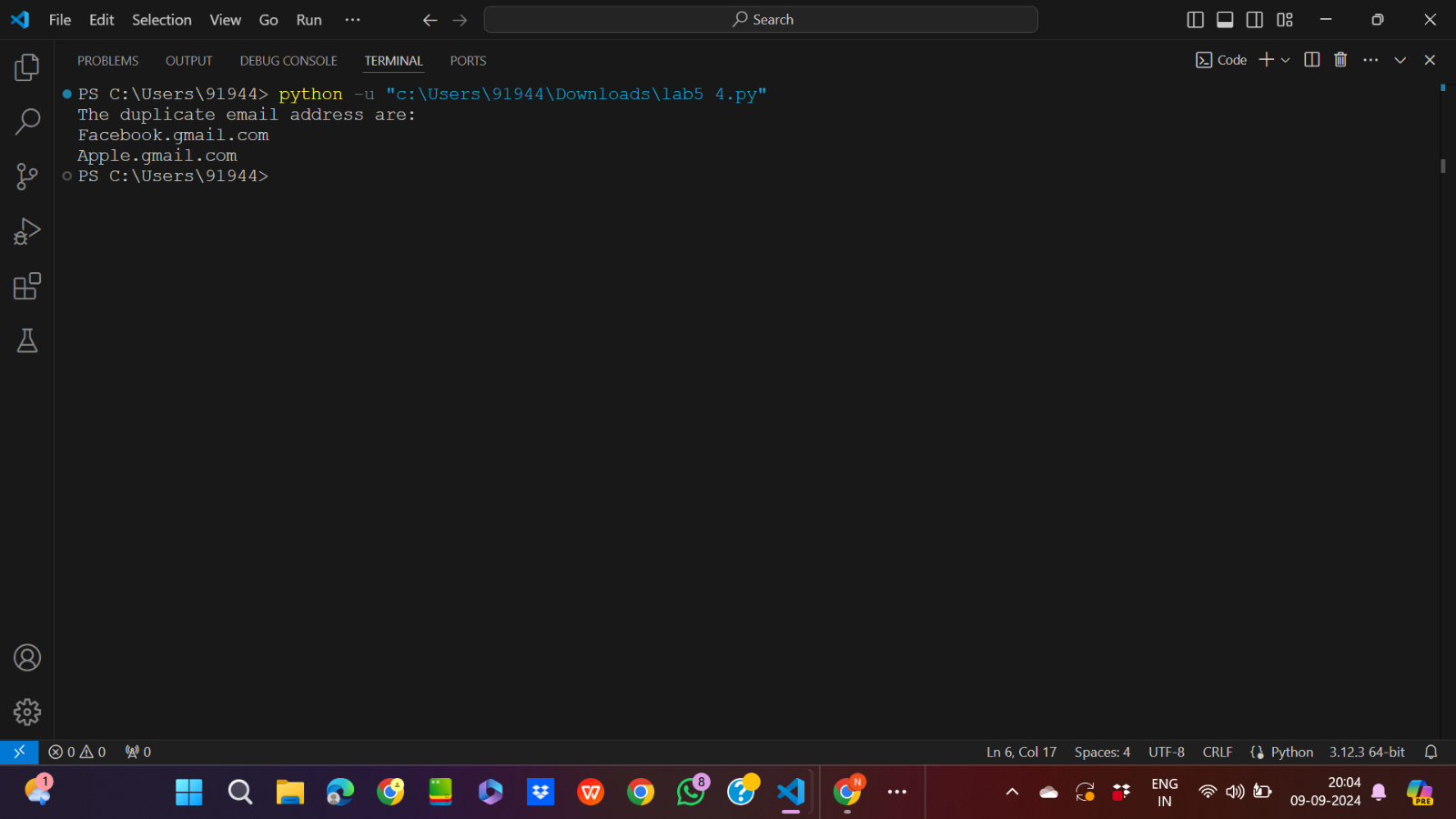
print("The duplicate email address are: ")

for i in set(email\_address):

    if(email\_address.count(i)>1):

        print(i)

Output:



5.

Input code:

myset1={"Apple","Banana","Cherry"}

myset2={"Apple","Banana","Cherry","Emli","Mango","Orange"}

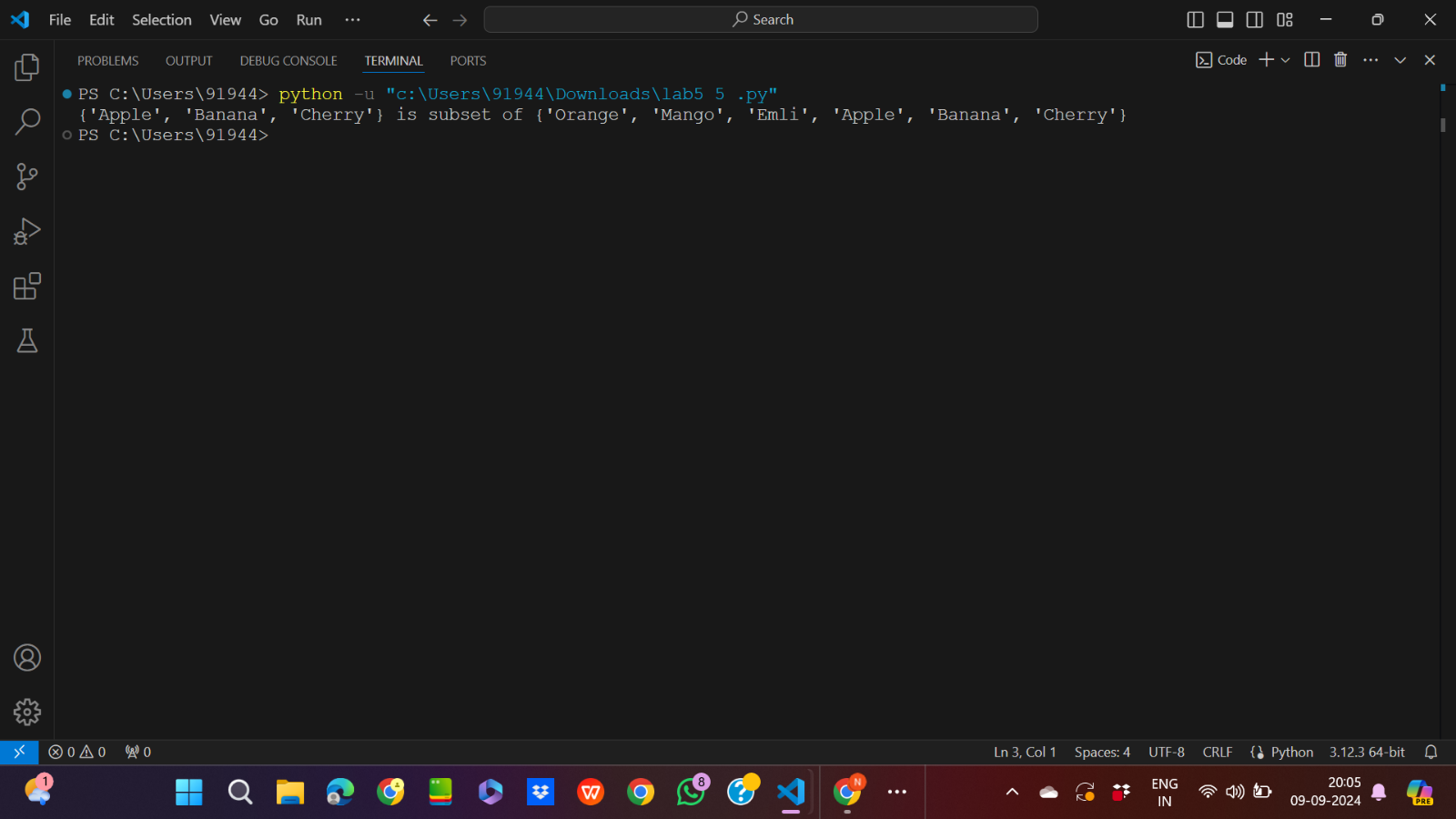
if myset1<=myset2:

    print(f"{myset1} is subset of {myset2}")

else:

    print(f"{myset1} is not subset of {myset2}")

Output:



6.

Input code:

doc1={"Apple","Papaya","Mango"}

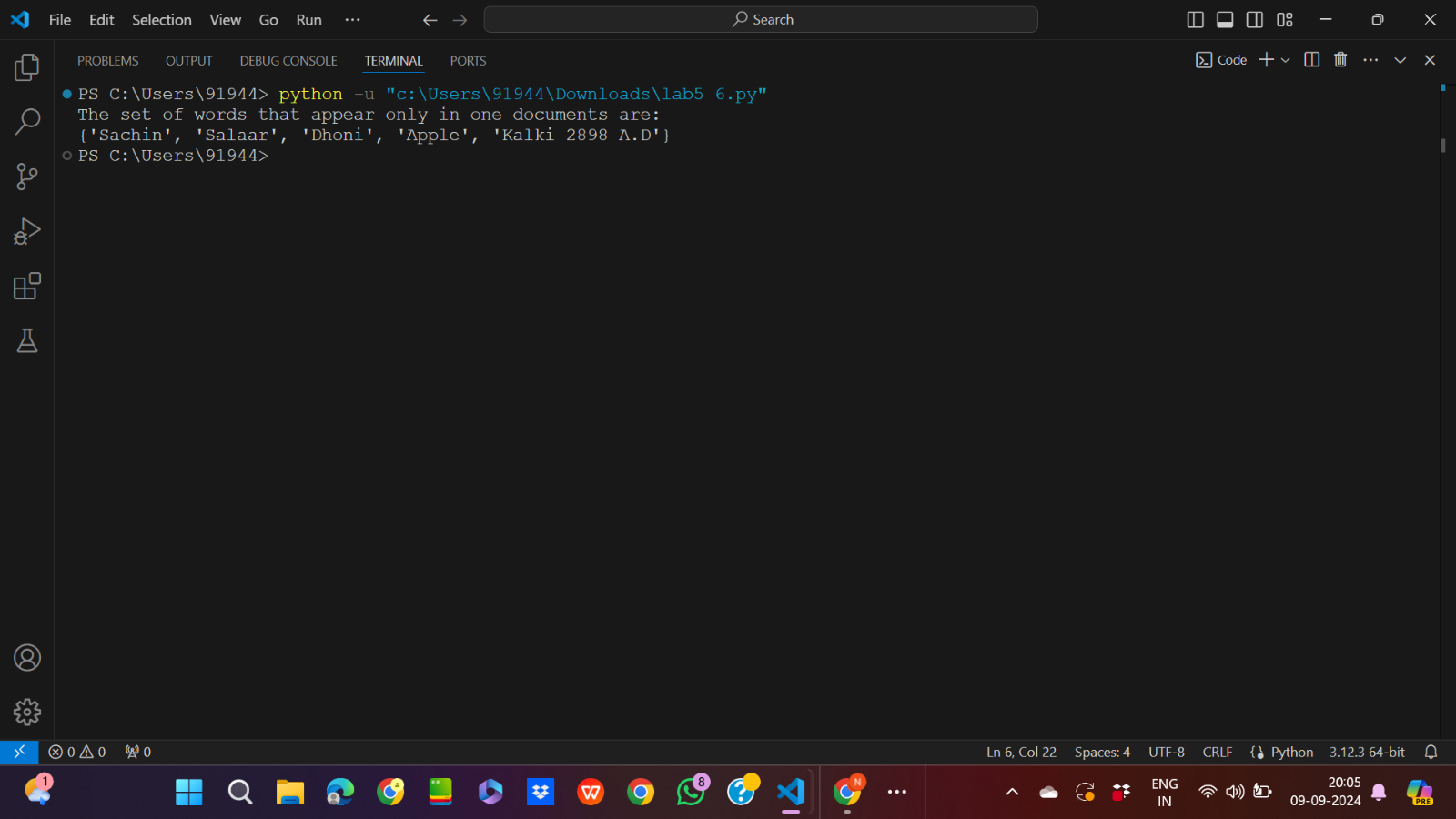
doc2={"Sachin","Papaya","Dhoni"}

doc3={"Kalki 2898 A.D","Salaar","Mango"}

print(f"The set of words that appear only in one documents are: ")

print(doc1^doc2^doc3)

Output:



7.

Input code:

def common(a,b):

    print(a&b)

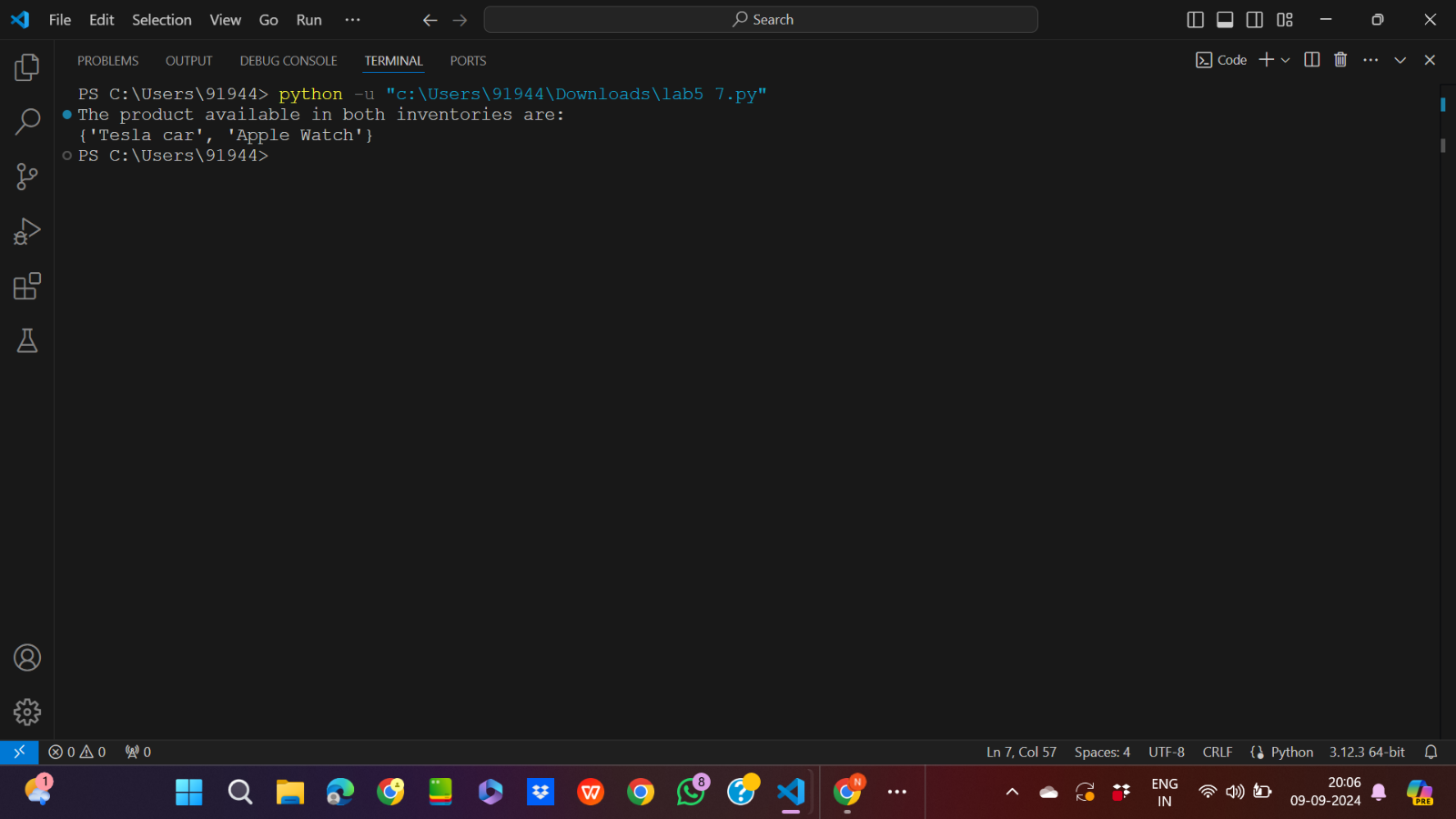
myset1={"Apple Watch","Tesla car","Costco gear cycle"}

myset2={"Apple Watch","Tesla car","Amazon Mobile"}

print("The product available in both inventories are: ")

common(myset1,myset2)

Output:



8.

Input code:

directed\_graph={"A":{"B","C"},"B":{"D"},"C":{"D"},"D":{"E"},"E":set()}

def DFS(graph, start, visited=None):

    if visited is None:

        visited = set()

    visited.add(start)

    for neighbor in graph.get(start, []):

        if neighbor not in visited:

            DFS(graph, neighbor, visited)

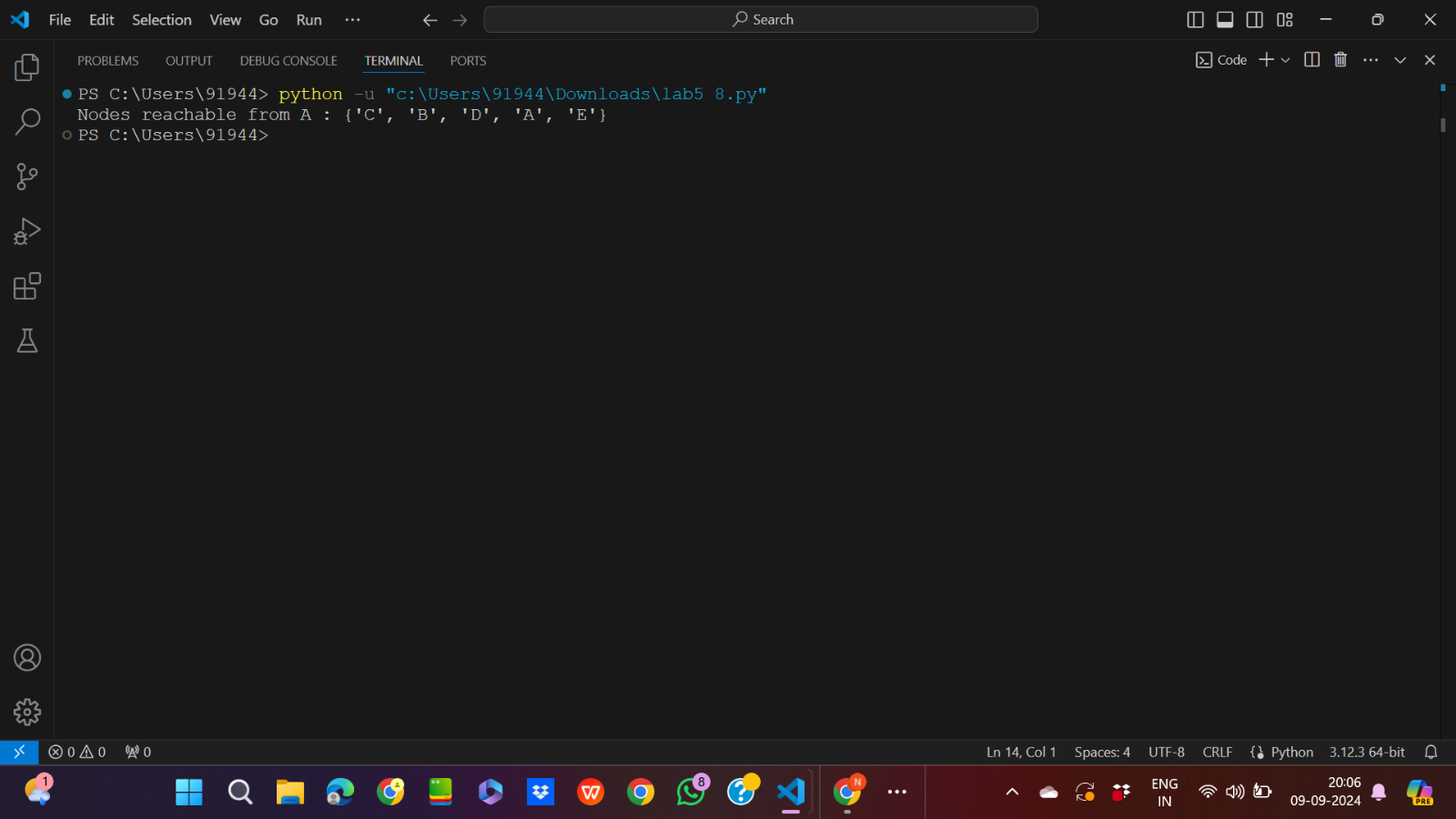
    return visited

start\_node = "A"

reachable\_nodes = DFS(directed\_graph, start\_node)

print("Nodes reachable from", start\_node, ":", reachable\_nodes)

Output:



9.

Input code:

def has\_cycle(graph):

    def dfs(node, visited, rec\_stack):

        visited.add(node)

        rec\_stack.add(node)

        for neighbor in graph.get(node, []):

            if neighbor not in visited and dfs(neighbor, visited, rec\_stack):

                return True

            elif neighbor in rec\_stack:

                return True

        rec\_stack.remove(node)

        return False

    visited = set()

    for node in graph:

        if node not in visited and dfs(node, visited, set()):

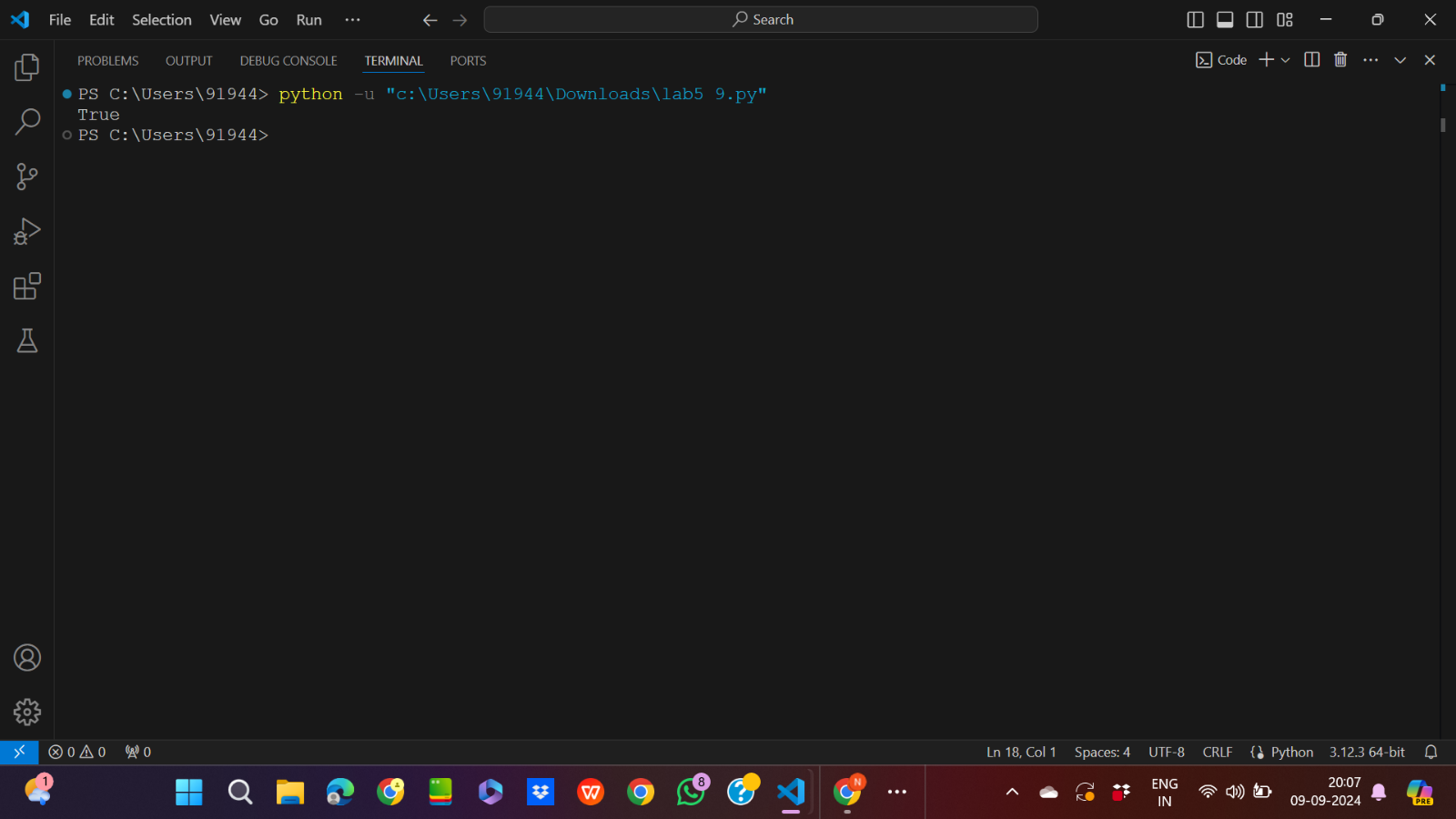
            return True

    return False

graph={"A":{"B"},"B":{"C"},"C":{"A"},"D":{"E"},"E":set()}

print(has\_cycle(graph))

Output:



10.

Input code:

a=input("Enter the element of set with space betweem them: ")

s=set(a.split(" "))

def powerset(s):

    s = list(s)

    result = [[]]

    for elem in s:

        result += [subset + [elem] for subset in result]

    return result

print(f"The powerset of {s} are: ")

print(powerset(s))

Output:

