# LAB ASSIGNMENT-10

TASK-1:

Prompt:

Identify and correct the code

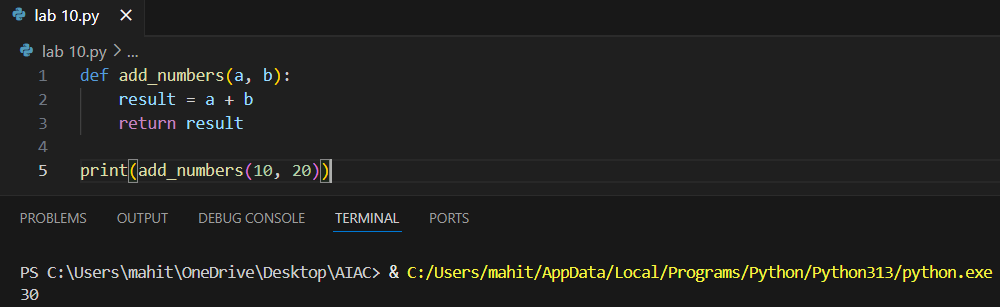
def add\_numbers(a, b)

result = a + b

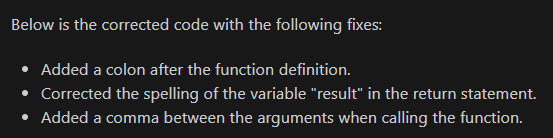
return reslt

print(add\_numbers(10 20))

Code and Output:



Code Explanation:



TASK-2:

Prompt:

Optimize inefficient logic while keeping the result correct.

def find\_duplicates(nums):

duplicates = []

for i in range(len(nums)):

for j in range(len(nums)):

if i != j and nums[i] == nums[j] and nums[i] not in duplicates:

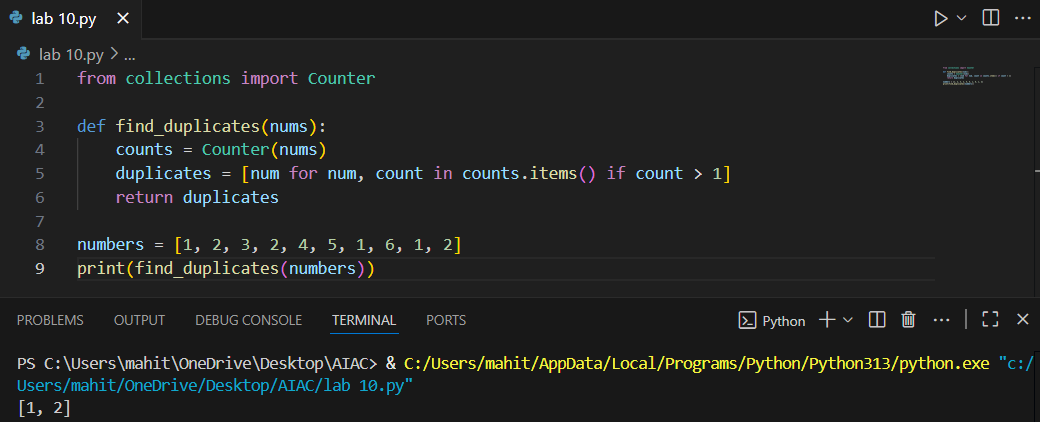
duplicates.append(nums[i])

return duplicates

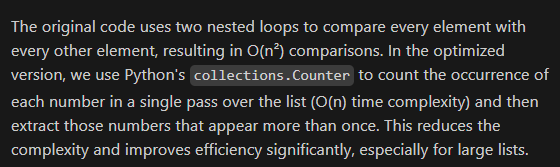
numbers = [1,2,3,2,4,5,1,6,1,2]

print(find\_duplicates(numbers))

Code and Output:



Code Explanation:



TASK-3:

Prompt:

Generate a Function renamed to calculate\_factorial.

Proper indentation, variable naming, docstrings, and formatting.

Provide a more readable version.

def c(n):

x=1

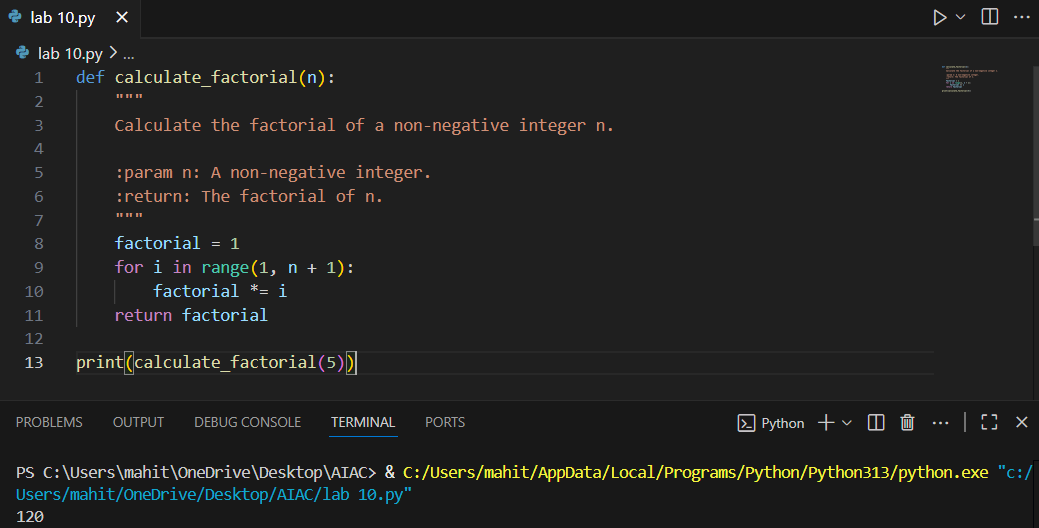
for i in range(1,n+1):

x=x\*i

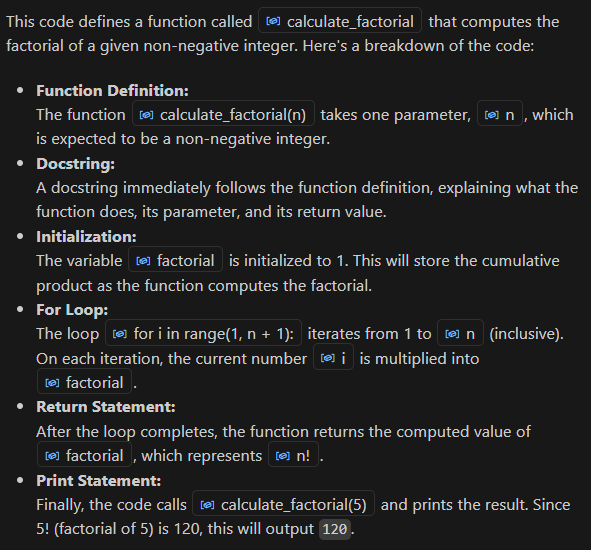
return x

print(c(5))

Code and Output:



Code Explanation:



TASK-4:

Prompt:

Add security practices and exception handling to the code.

import sqlite3

def get\_user\_data(user\_id):

conn = sqlite3.connect("users.db")

cursor = conn.cursor()

query = f"SELECT \* FROM users WHERE id = {user\_id};" # Potential SQL injection risk

cursor.execute(query)

result = cursor.fetchall()

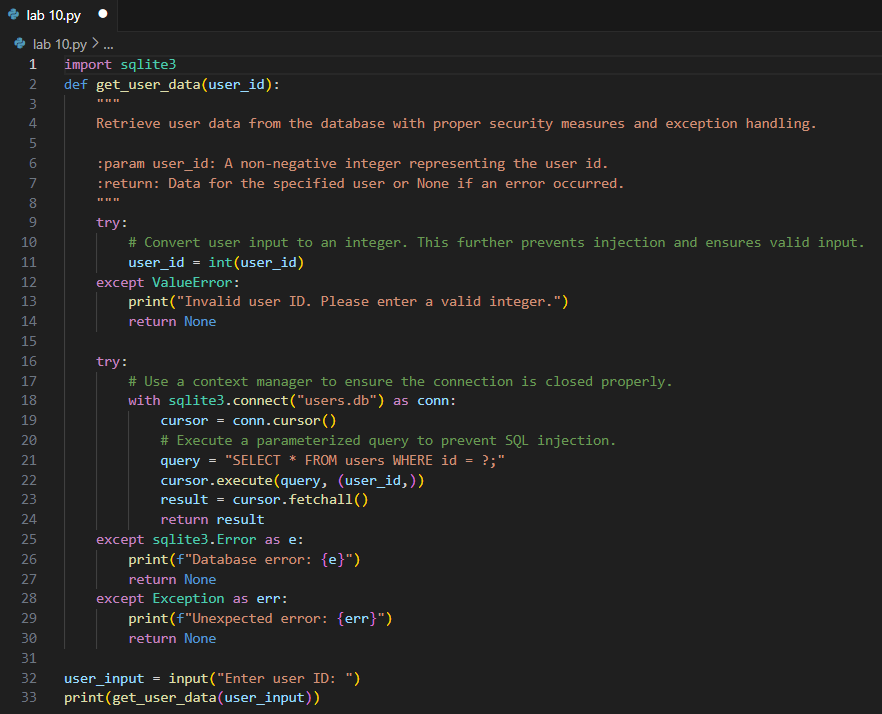
conn.close()

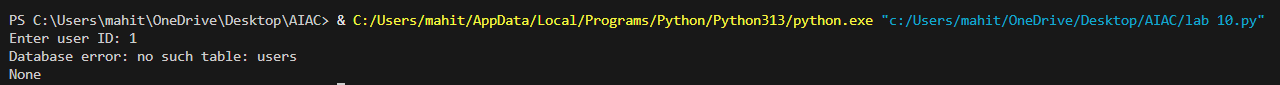
return result

user\_input = input("Enter user ID: ")

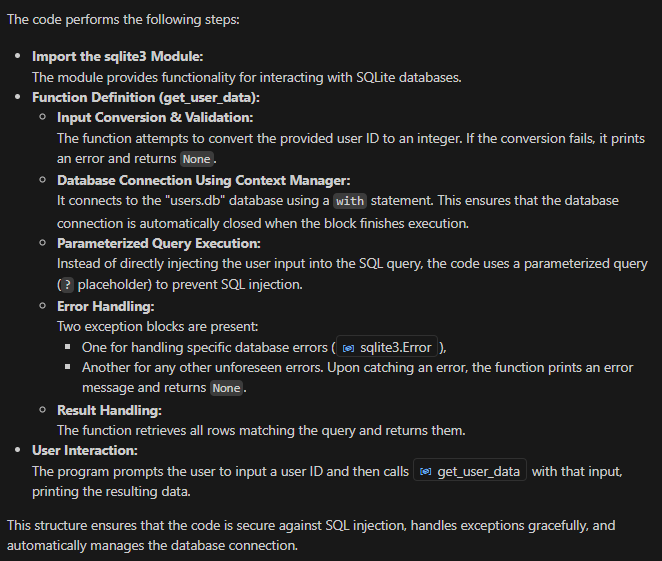
print(get\_user\_data(user\_input))

Code and Output:





Code Explanation:



TASK-5:

Prompt:

Generate a review report for this messy code.

def calc(x,y,z):

if z=="add":

return x+y

elif z=="sub": return x-y

elif z=="mul":

return x\*y

elif z=="div":

return x/y

else: print("wrong")

print(calc(10,5,"add"))

print(calc(10,0,"div"))

Output:

