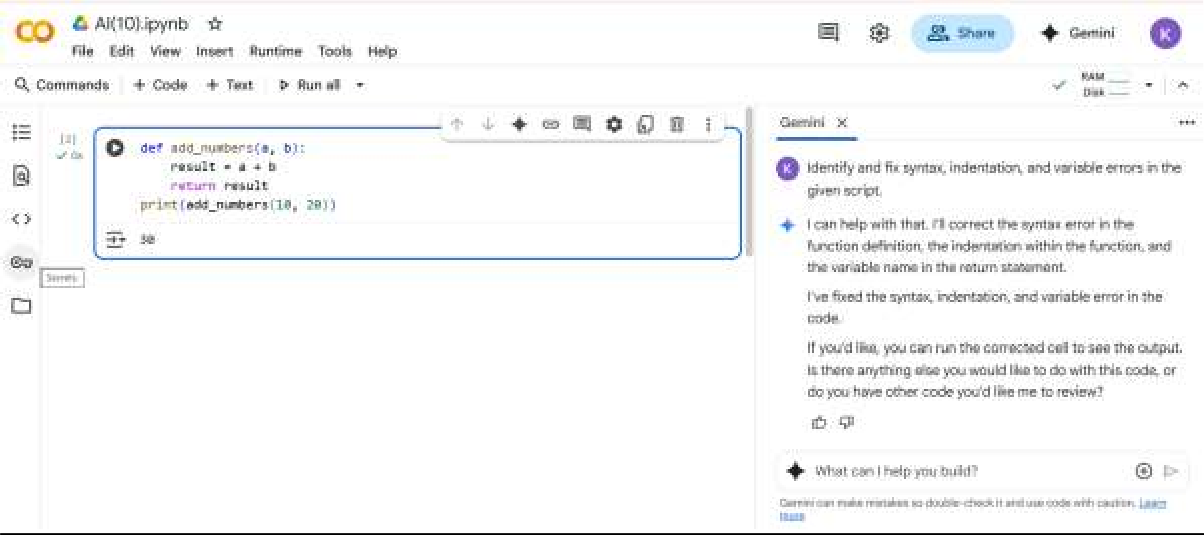
Assignment-10

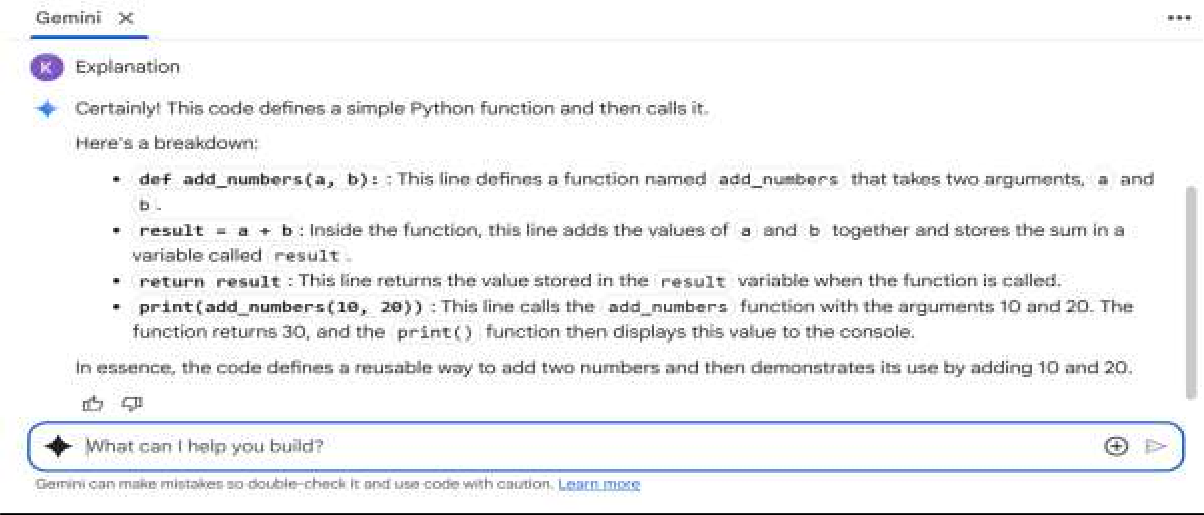
Task 1:

Identify and fix syntax, indentation, and variable errors in the given script. # buggy\_code\_task1.py

def add\_numbers(a, b) result = a + b

return reslt print(add\_numbers(10 20)) code and output:





Task 2:

Optimize inefficient logic while keeping the result correct. # buggy\_code\_task2.py

def find\_duplicates(nums): duplicates = []

for i in range(len(nums)): for j in range(len(nums)):

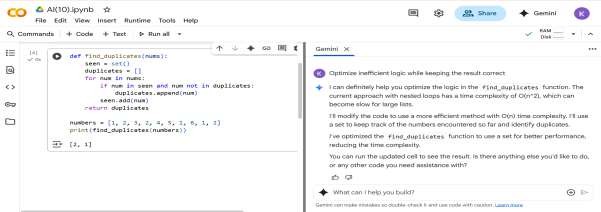
if i != j and nums[i] == nums[j] and nums[i] not in duplicates: duplicates.append(nums[i])

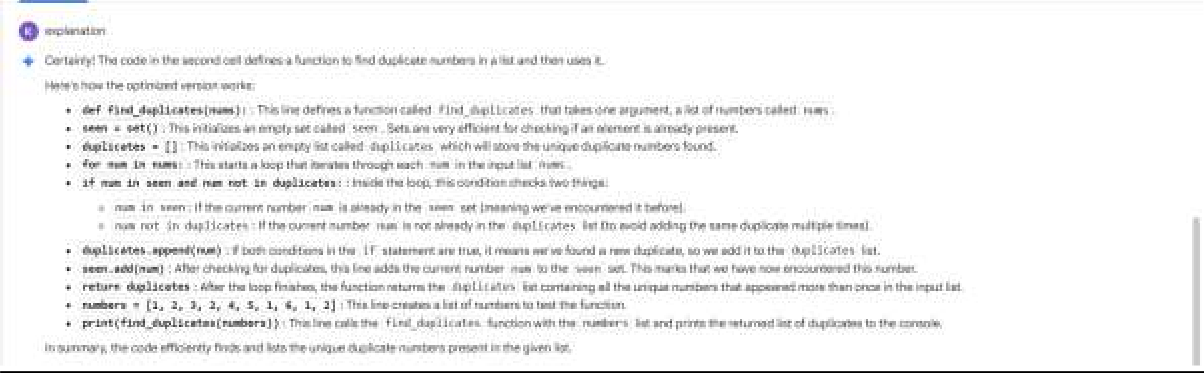
return duplicates

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

print(find\_duplicates(numbers))

Code and output:





Task 3:

Refactor messy code into clean, PEP 8–compliant, well-structured code. # buggy\_code\_task3.py

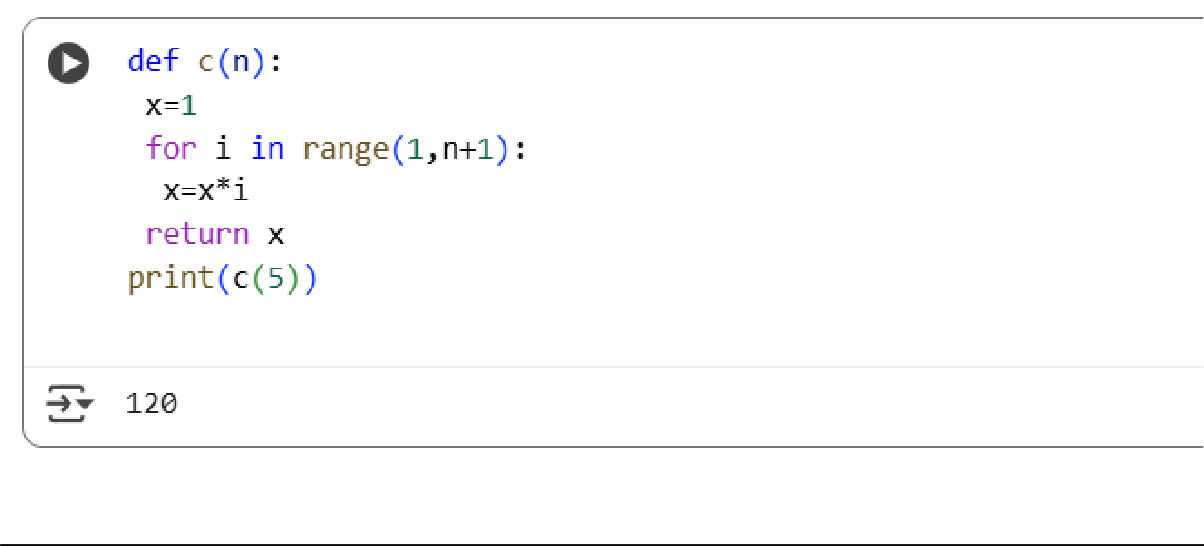
def c(n):

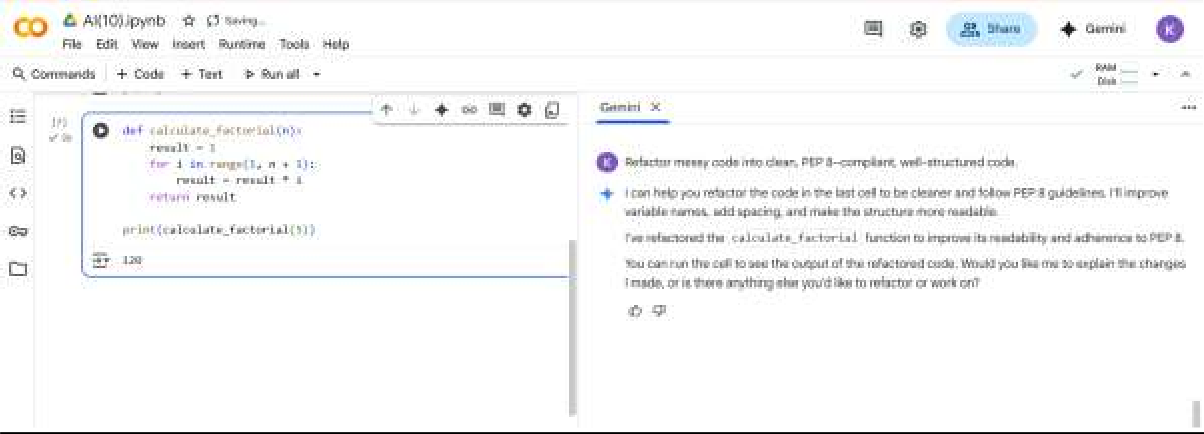
x=1

for i in range(1,n+1): x=x\*i

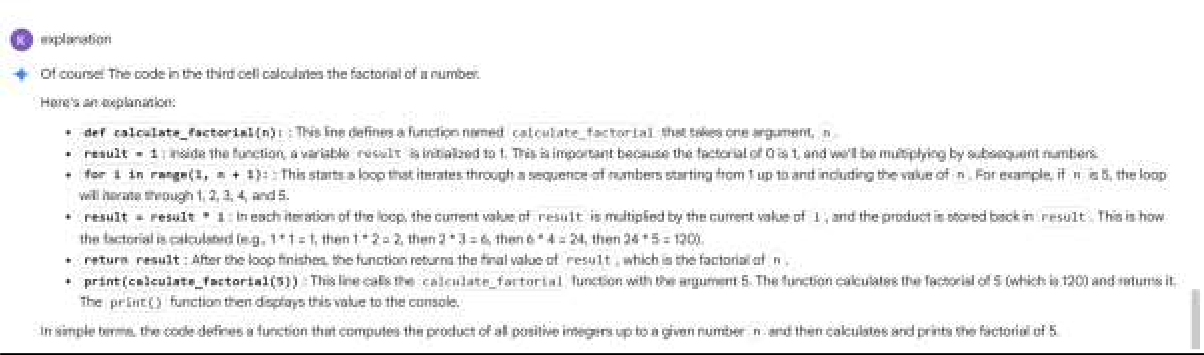
return x print(c(5))

code and output:





Explanation:



Task 4:

Add security practices and exception handling to the code. # buggy\_code\_task4.py

import sqlite3

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

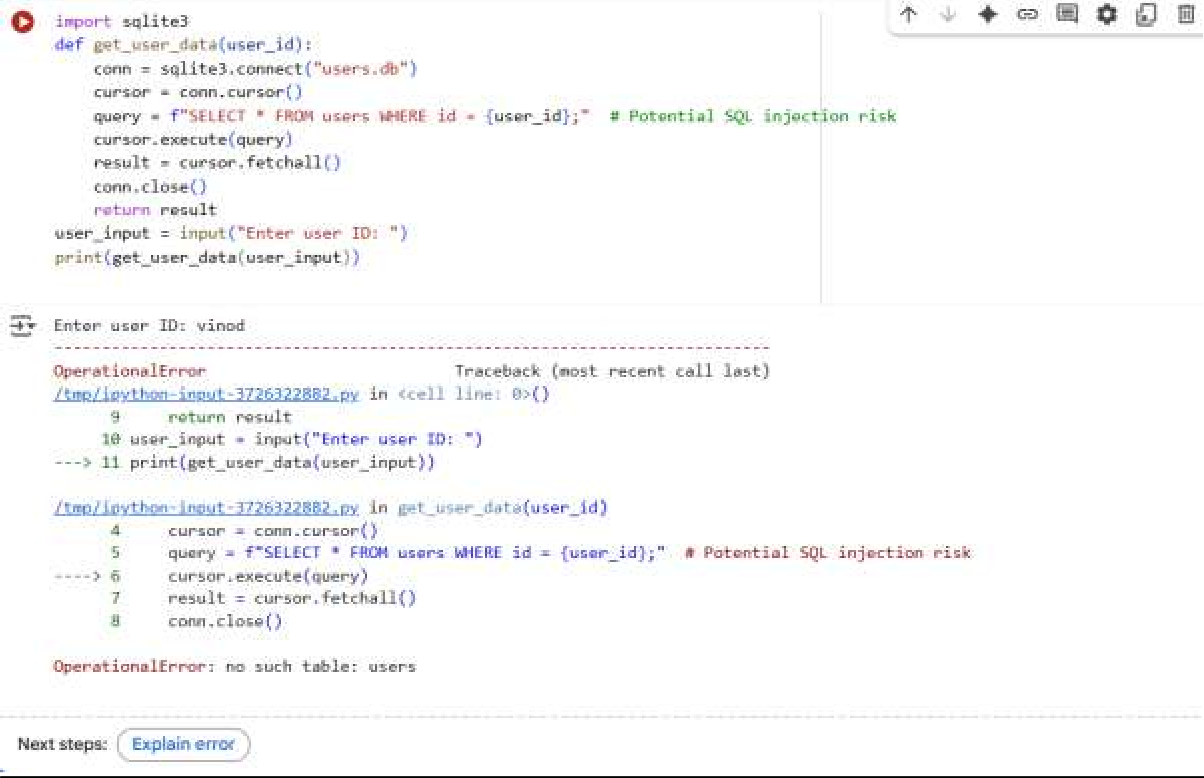
conn = sqlite3.connect("users.db") cursor = conn.cursor()

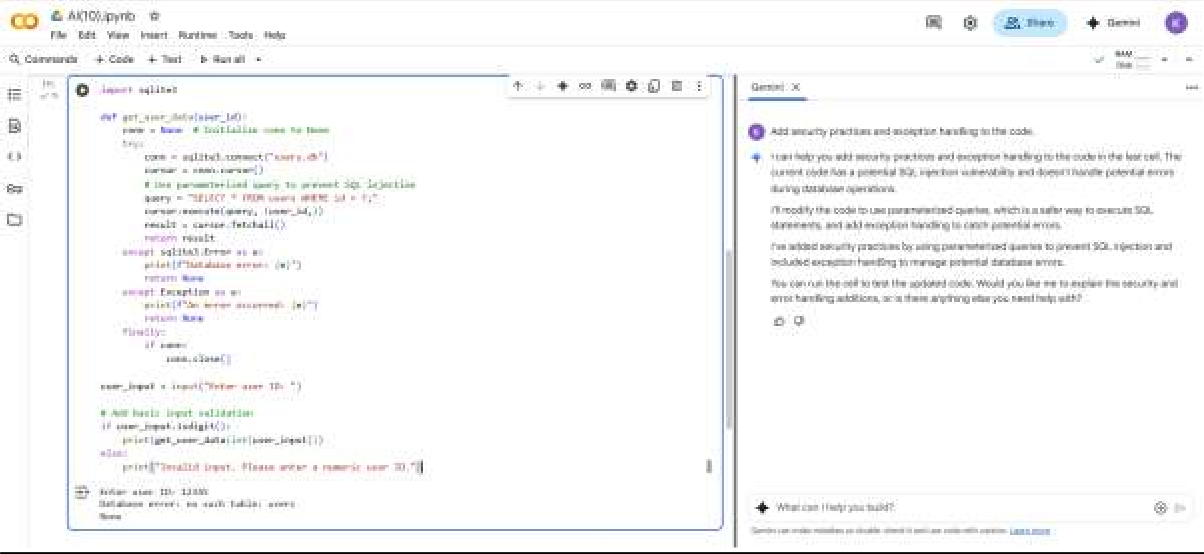
query = f"SELECT \* FROM users WHERE id = {user\_id};" # Potential SQL injection risk cursor.execute(query)

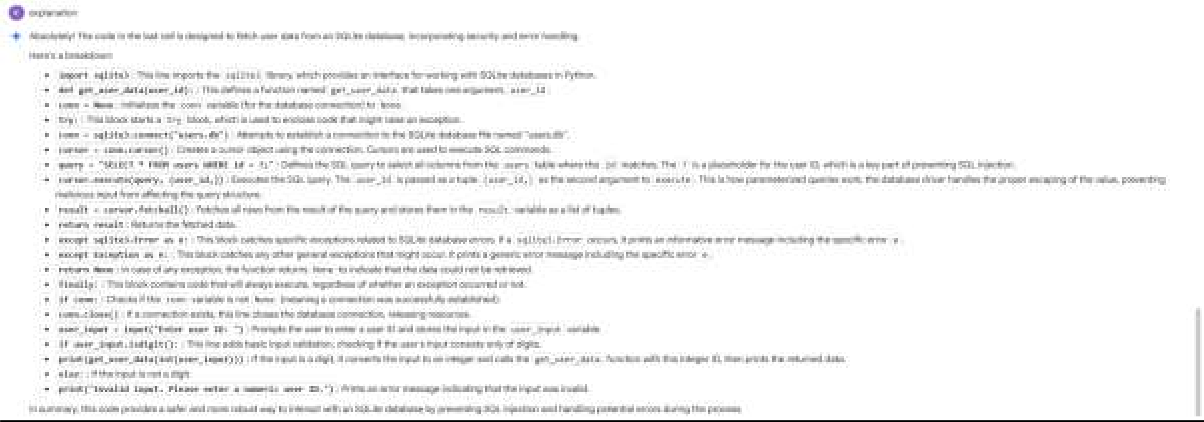
result = cursor.fetchall() conn.close()

return result

user\_input = input("Enter user ID: ") print(get\_user\_data(user\_input))







Task 5:

Generate a review report for this messy code. # buggy\_code\_task5.py

def calc(x,y,z):

if z=="add": return x+y

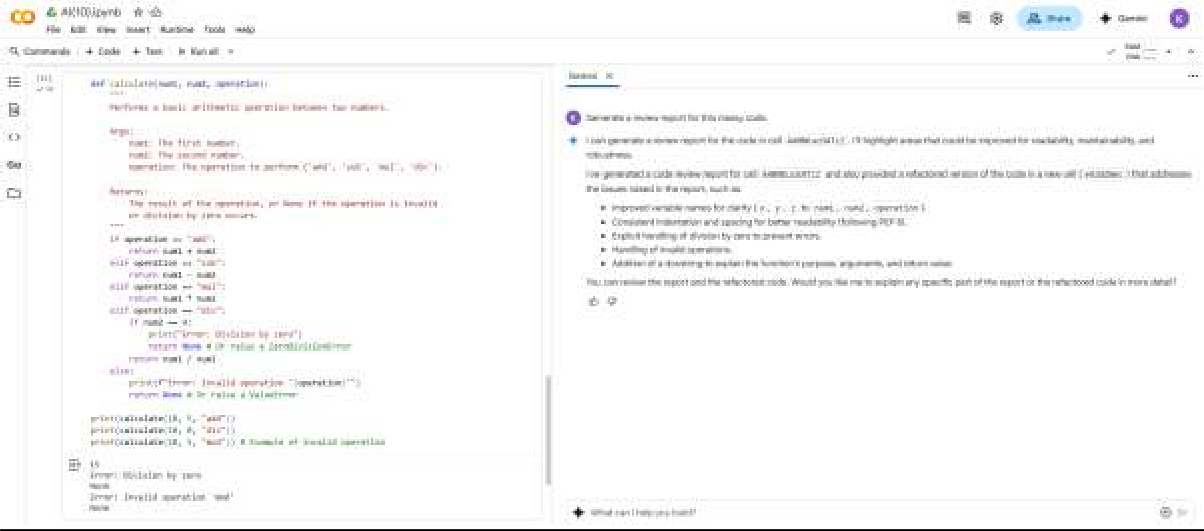
elif z=="sub": return x-y elif z=="mul":

return x\*y elif z=="div": return x/y

else: print("wrong") print(calc(10,5,"add"))

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





Explanation:

