AI ASSISSTED CODING

ASSIGNMENT-6.1

N. Madhuvani

2303A52117

Batch-40

Task Description #1 (AI-Based Code Completion for Loops)

Task: Use an AI code completion tool to generate a loop-based

program.

Prompt:

“Generate Python code to print all even numbers between 1 and N

using a loop.”

Expected Output:

• AI-generated loop logic.

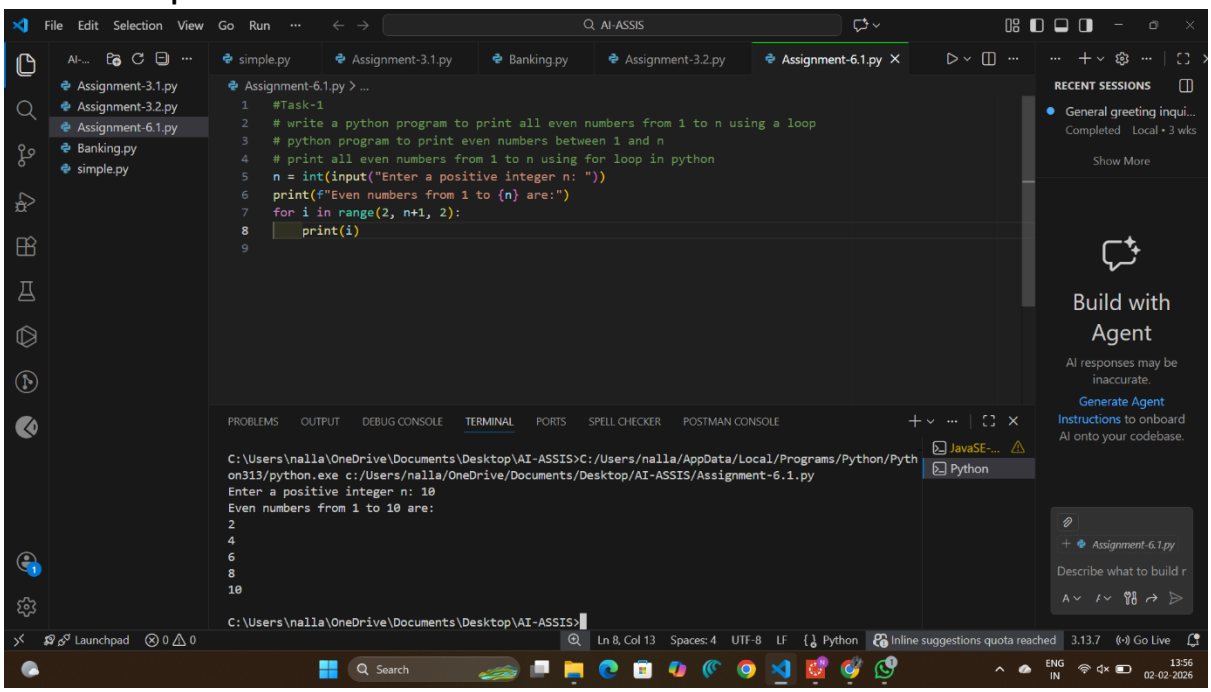
• Identification of loop type used (for or while).

• Validation with sample inputs.

Prompt:

write a python program to print all even numbers from 1 to n using a loop

Code and Output:



Analysis:

1. The program takes a positive integer n as input from the user.
2. A for loop is used to iterate from 2 to n.
3. The loop increments by 2, so only even numbers are printed.
4. This approach avoids unnecessary conditional checks.
5. The program demonstrates the use of loops and user input.

Task Description #2 (AI-Based Code Completion for Loop with Conditionals)

Task: Use an AI code completion tool to combine loops and conditionals.

Prompt:

“Generate Python code to count how many numbers in a list are even and odd.”

Expected Output:

• AI-generated code using loop and if condition.

• Correct count validation.

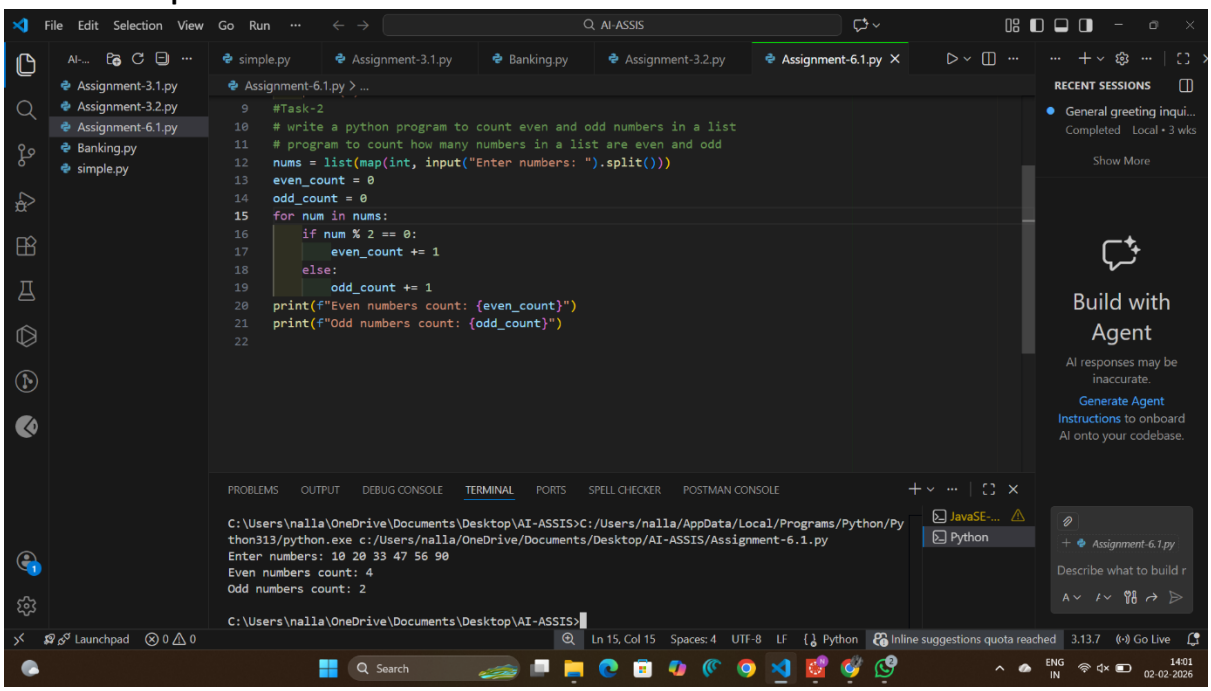
• Explanation of logic flow.

Prompt:

write a python program to count even and odd numbers in a list

program to count how many numbers in a list are even and odd

Code and Output:



Analysis:

1. The program accepts multiple numbers from the user as a list.
2. Two counters are initialized to count even and odd numbers.
3. A for loop iterates through each number in the list.
4. The modulus operator (%) is used to check even or odd.
5. The final count of even and odd numbers is displayed.

Task Description #3 (AI-Based Code Completion for Class

Attributes Validation)

Task: Use an AI tool to complete a Python class that validates user

input.

Prompt:

“Generate a Python class User that validates age and email using

conditional statements.”

Expected Output:

• AI-generated class with validation logic.

• Verification of condition handling.

• Test cases for valid and invalid inputs.

Prompt:

#create a python class named User with attributes name, age, and email

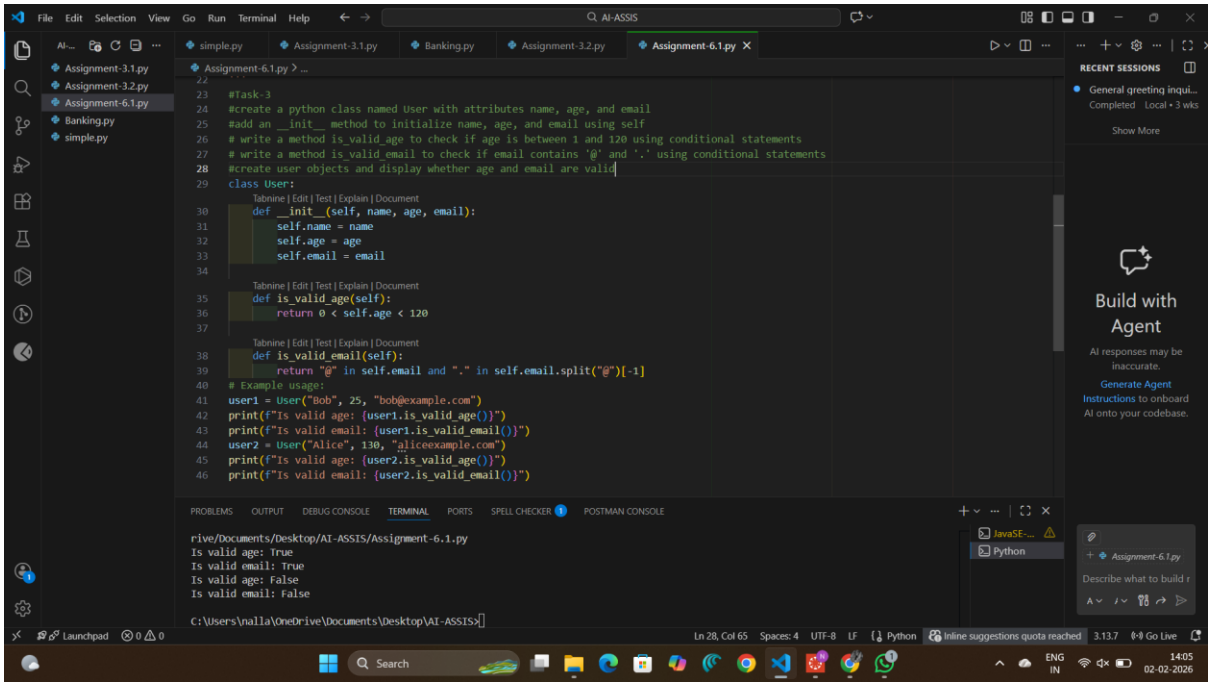
#add an __init__ method to initialize name, age, and email using self

write a method is_valid_age to check if age is between 1 and 120 using conditional statements

write a method is_valid_email to check if email contains '@' and '.' using conditional statements

#create user objects and display whether age and email are valid

Code and Output:



Analysis:

1. A User class is created with name, age, and email attributes.
2. The constructor initializes user details using the self keyword.
3. Age validation checks whether the age lies between 1 and 120.
4. Email validation checks for the presence of @ and ..
5. Objects are created to test valid and invalid user data.

Task Description #4 (AI-Based Code Completion for Classes)

Task: Use an AI code completion tool to generate a Python class for

managing student details.

Prompt:

“Generate a Python class Student with attributes (name, roll number,

marks) and methods to calculate total and average marks.”

Expected Output:

• AI-generated class code.

• Verification of correctness and completeness of class

structure.

• Minor manual improvements (if needed) with justification.

Prompt:

#write a python program and create a python class named Student with attributes name, roll_number, and marks

#add an __init__ method to initialize name, roll_number, and marks (marks should be a list)

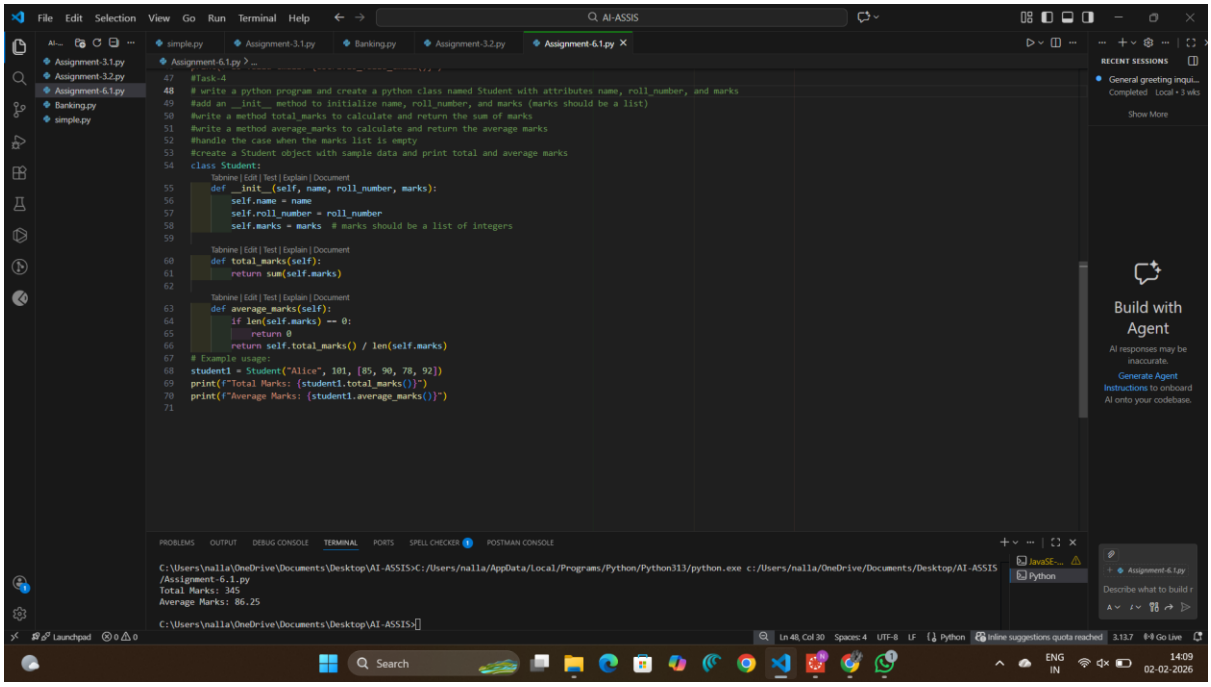
#write a method total_marks to calculate and return the sum of marks

#write a method average_marks to calculate and return the average marks

#handle the case when the marks list is empty

#create a Student object with sample data and print total and average marks

Code and Output:



Analysis:

1. A Student class is defined with name, roll number, and marks.

2. Marks are stored as a list of integers.

3. The total_marks() method calculates the sum of marks.

4. The average_marks() method calculates the average safely.

5. A condition handles the case when the marks list is empty.

Task Description 5 (AI-Assisted Code Completion Review)

Task: Use an AI tool to generate a complete Python program using

classes, loops, and conditionals together.

Prompt:

“Generate a Python program for a simple bank account system using

class, loops, and conditional statements.”

Expected Output:

• Complete AI-generated program.

• Identification of strengths and limitations of AI suggestions.

• Reflection on how AI assisted coding productivity.

Prompt:

#Write a python code and create a python class named BankAccount with attributes account_holder and balance

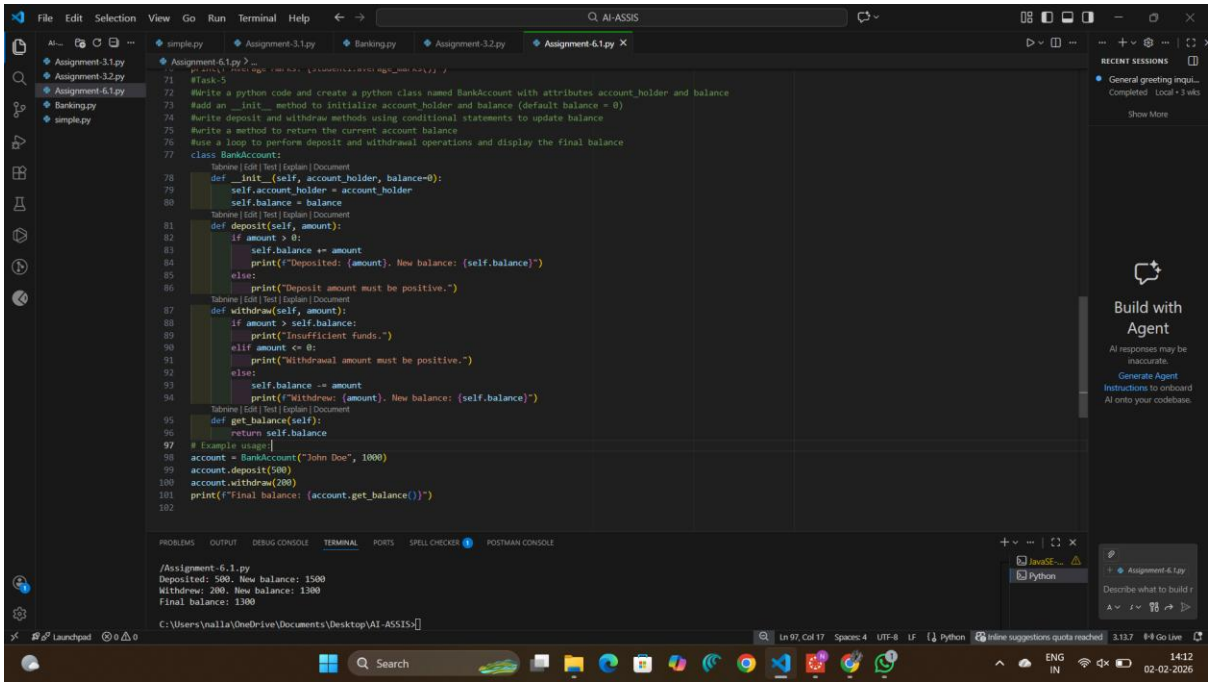
#add an __init__ method to initialize account_holder and balance (default balance = 0)

#write deposit and withdraw methods using conditional statements to update balance

#write a method to return the current account balance

#use a loop to perform deposit and withdrawal operations and display the final balance

Code and Output:



Analysis:

1. A BankAccount class is created with account holder and balance.
2. The deposit method adds money only if the amount is positive.
3. The withdraw method checks for sufficient balance and valid amount.
4. Conditional statements ensure safe banking operations.
5. The program displays the final account balance after transactions.