ASSIGNMENT – 4

NAME: swaranjith reddy thatipalli

HT.NO: 2403A52049

BATCH NO: AIB03

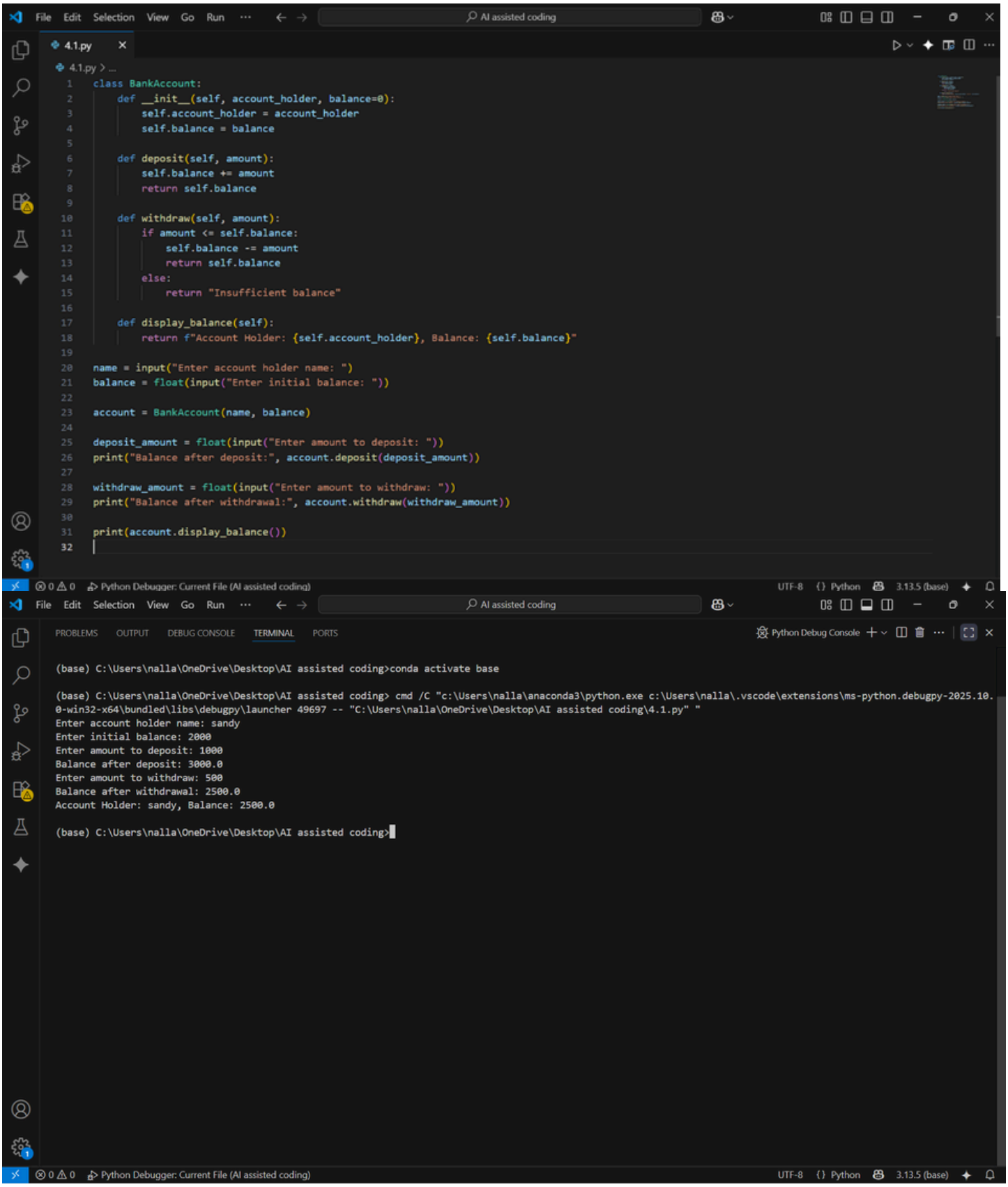
TASK 1:

Complete class with methods like:deposit(self, amount)withdraw(self, amount)display_balance(self)

PROMPT:

Complete class with methods like:deposit(self, amount)withdraw(self, amount)display_balance(self)

CODE:



OBSERVATION:

This code defines a BankAccount class to manage a simple bank account with a holder's name and balance. It has methods to deposit money, withdraw money if enough balance is available, and display account details. The program asks the user for their name, initial balance, deposit, and withdrawal amounts, updates the account accordingly, and shows the final balance.

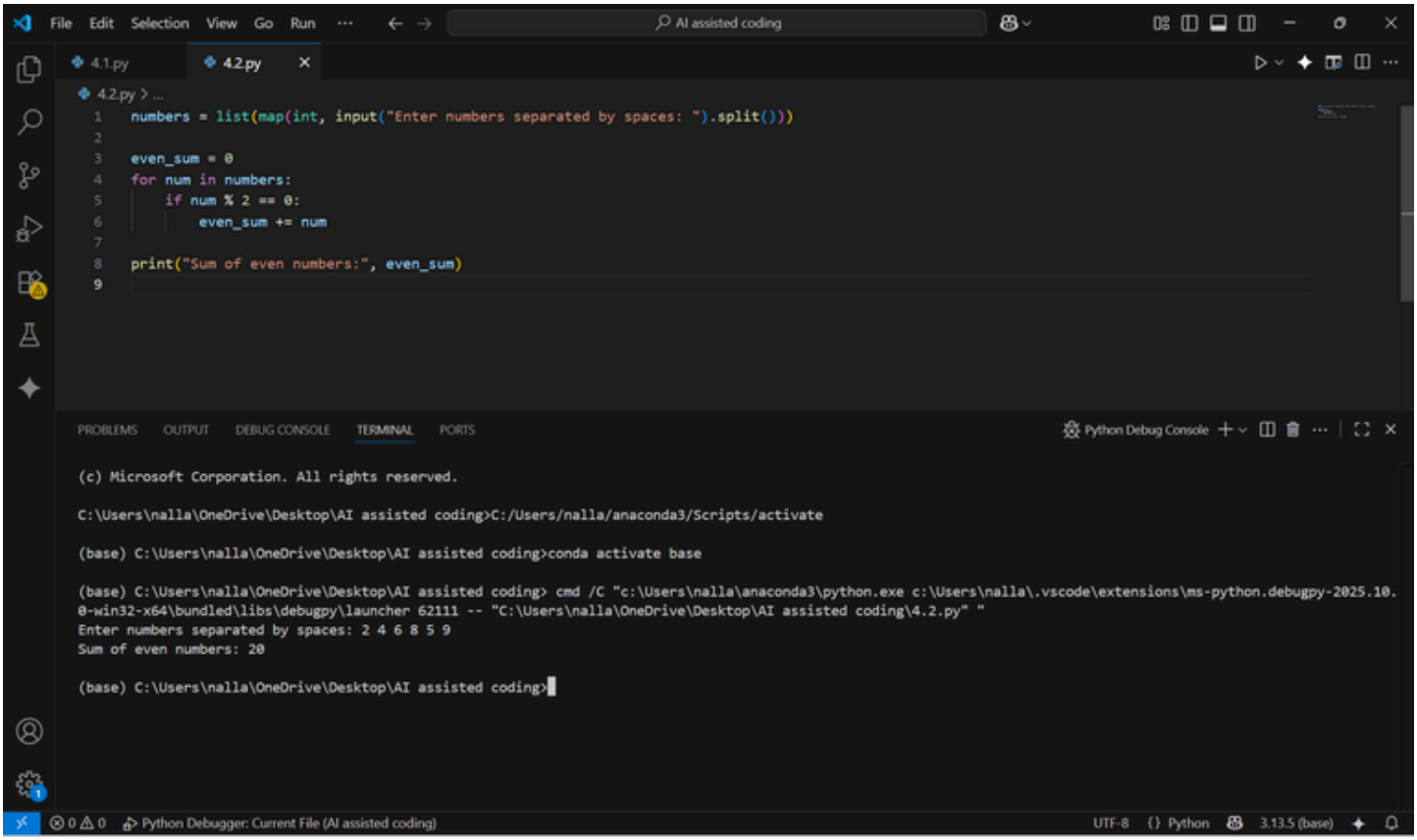
TASK 2:

Write a comment and the initial line of a loop to iterate over a list. Allow GitHub Copilot to complete the logic to sum all even numbers in the list.

PROMPT:

Write a comment and the initial line of a loop to iterate over a list. complete the logic to sum all even numbers in the list.

CODE:



OBSERVATION:

This Python code calculates the sum of even numbers from a list entered by the user. It first prompts the user to input numbers separated by spaces, then converts that input into a list of integers using map and split. It initializes a variable even_sum to zero, then iterates through each number in the list. If a number is divisible by 2 (i.e., it's even), it adds that number to even_sum. Finally, it prints the total sum of all even numbers entered by the user.

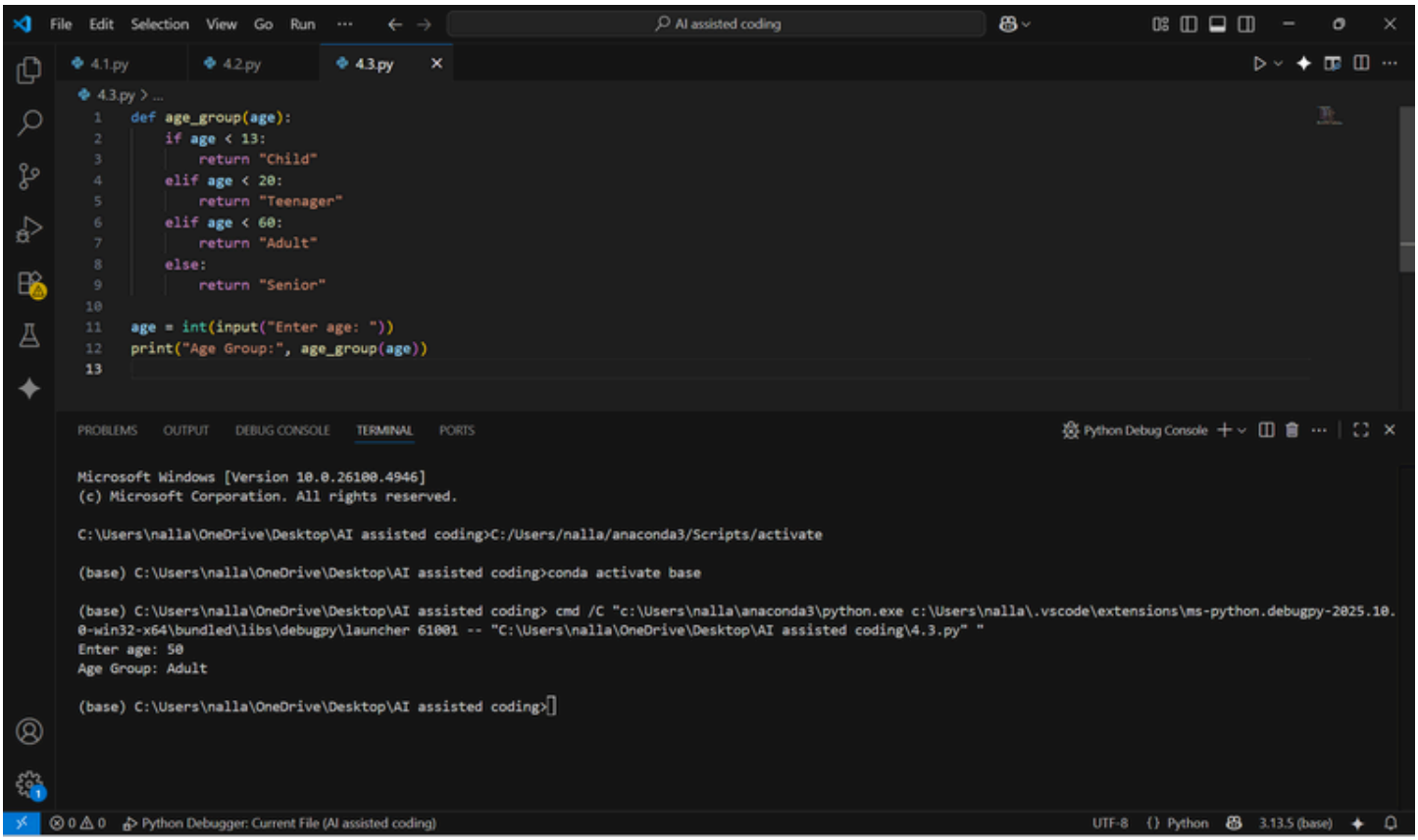
TASK 3:

Start a function that takes age as input and returns whether the person is a child, teenager, adult, or senior using if-elif-else. Use Copilot to complete the conditionals.

PROMPT:

Start a function that takes age as input and returns whether the person is a child, teenager, adult, or senior using if-elif-else**.**

CODE:



OBSERVATION:

This Python code defines a function age_group that categorizes a person based on their age. It takes an integer age as input and checks several conditions: if the age is less than 13, it returns "Child"; if it is between 13 and 19, it returns "Teenager"; if it is between 20 and 59, it returns "Adult"; and if it is 60 or above, it returns "Senior". The program prompts the user to enter their age, converts it to an integer, passes it to the age_group function, and then prints the resulting age category.

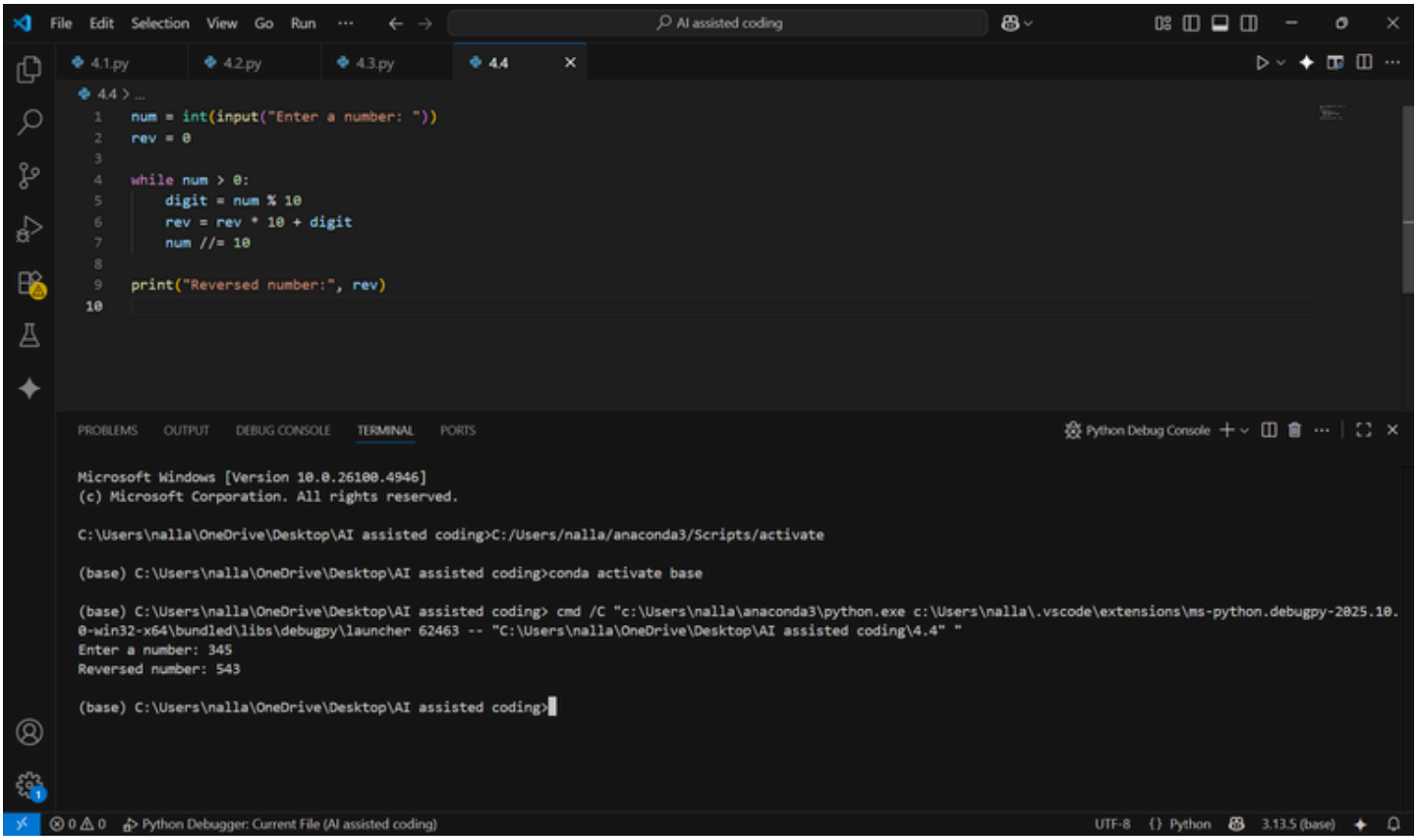
TASK 4:

Write a comment and start a while loop to reverse the digits of a number. Let Copilot completethe loop logic

PROMPT:

Write a comment and start a while loop to reverse the digits of a number.

CODE:



OBSERVATION:

This Python code reverses the digits of a number entered by the user. It first prompts the user to input a number and stores it in num, while initializing rev to 0 to hold the reversed number. Inside a while loop that runs as long as num is greater than 0, it extracts the last digit using num % 10, adds it to rev after shifting the existing digits to the left (rev * 10 + digit), and removes the last digit from num using integer division num //= 10. After the loop finishes, it prints the reversed number stored in rev.

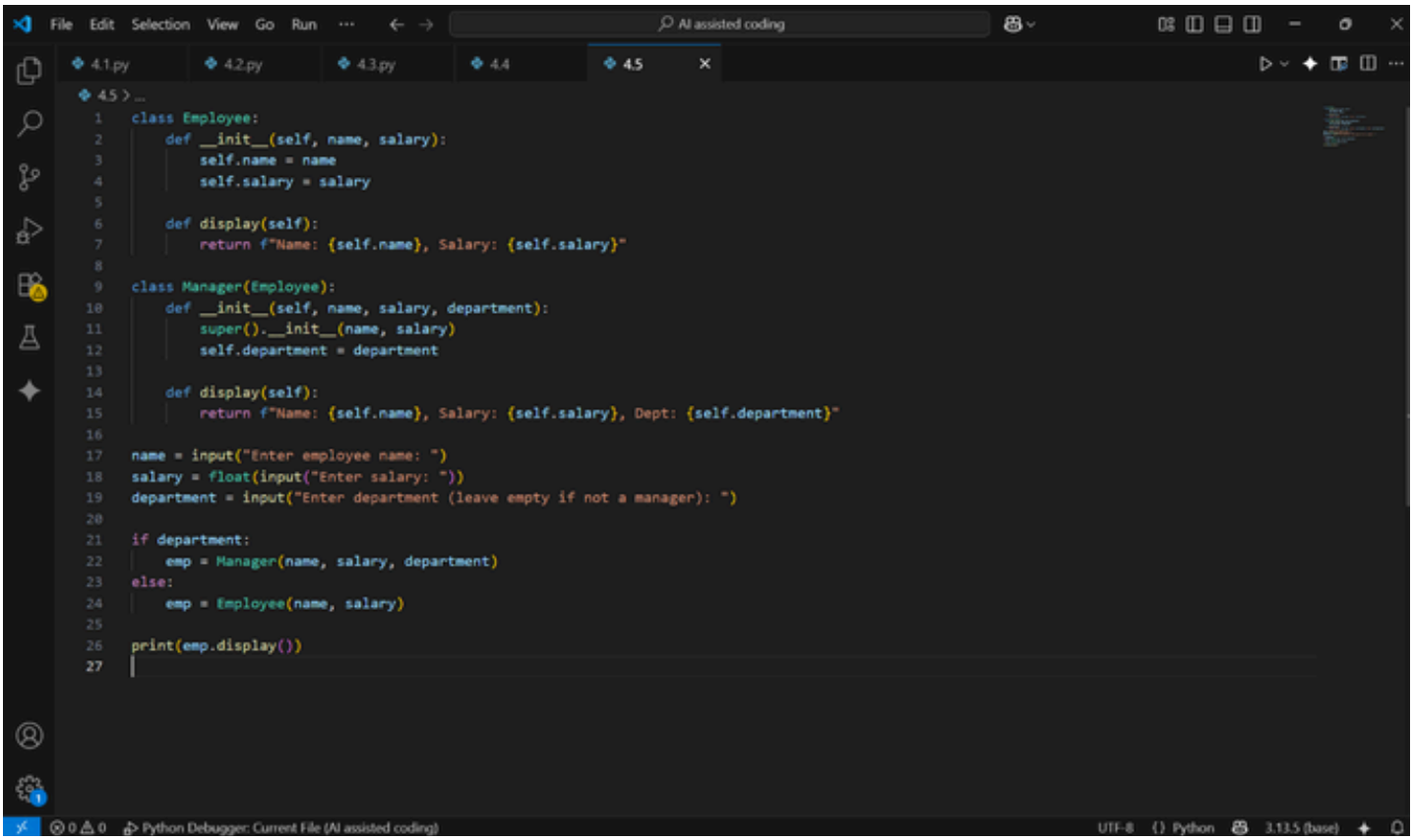
TASK 5:

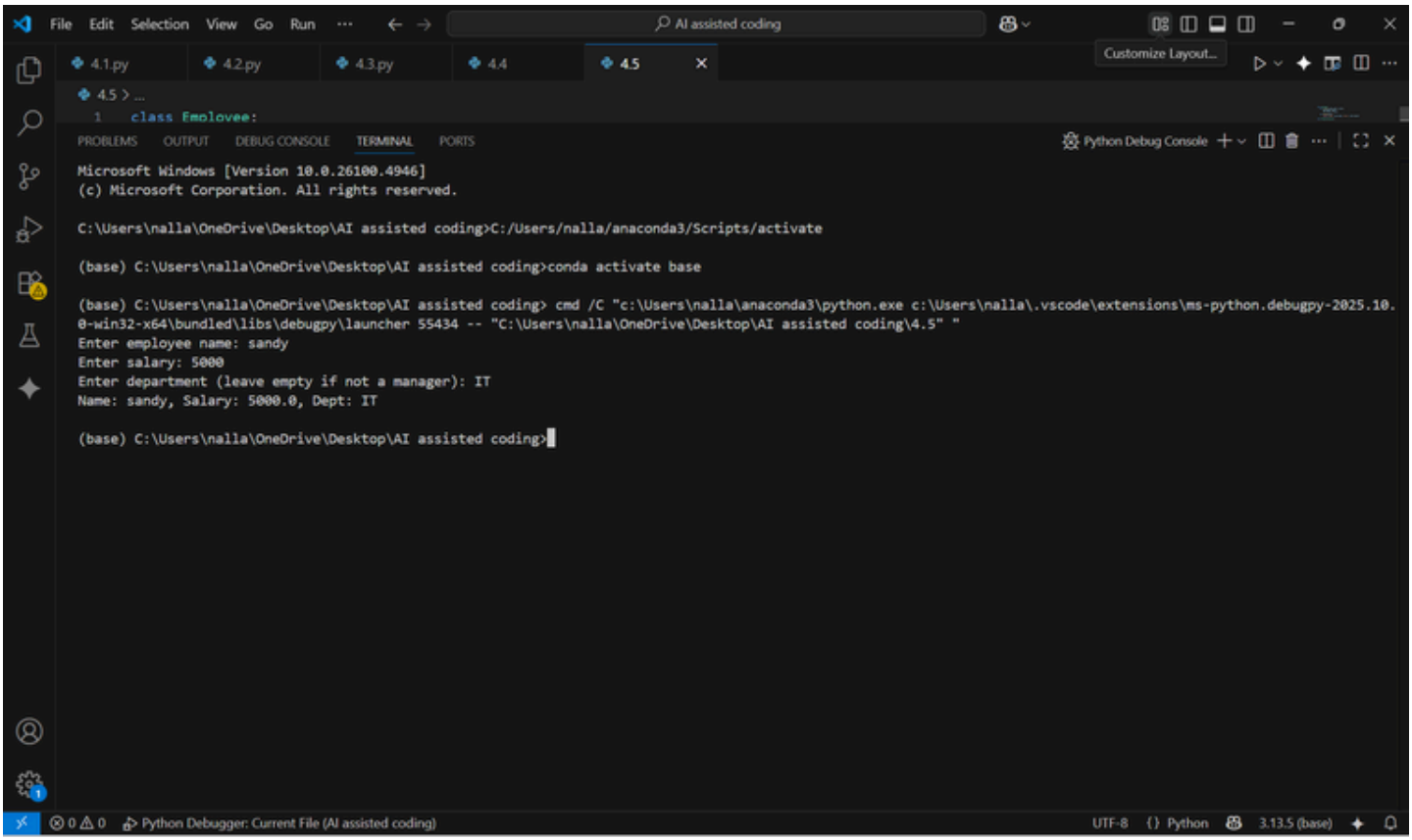
Begin a class Employee with attributes name and salary. Then, start a derived class Manager that inherits from Employee and adds a department. Let GitHub Copilot complete the methodsand constructor chaining

PROMPT:

Begin a class Employee with attributes name and salary. Then, start a derived class Manager that inherits from Employee and adds a department

CODE:





OBSERVATION:

This Python code demonstrates **inheritance** in classes. It defines an Employee class with a constructor that stores the employee's name and salary, and a display method that returns these details as a string. Then it defines a Manager class that inherits from Employee and adds a department attribute. The Manager class also overrides the display method to include the department in the output. The program asks the user to enter a name, salary, and optionally a department. If a department is provided, it creates a Manager object; otherwise, it creates a regular Employee object. Finally, it prints the details of the created object using the display method.