

**AI ASSISTED CODING**

**LAB-4:** ***AI-Based Code Auto-Completion – Classes, Loops, and Conditionals in Python using GitHub Copilot***

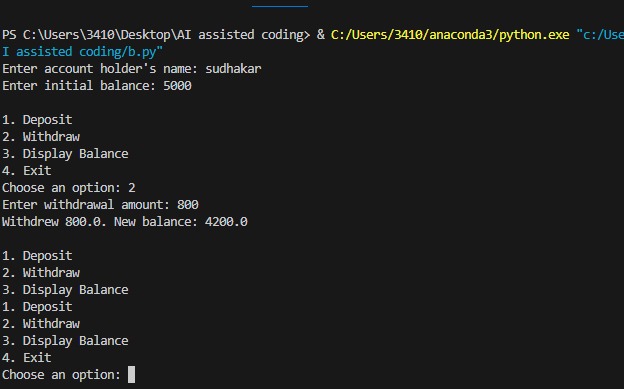
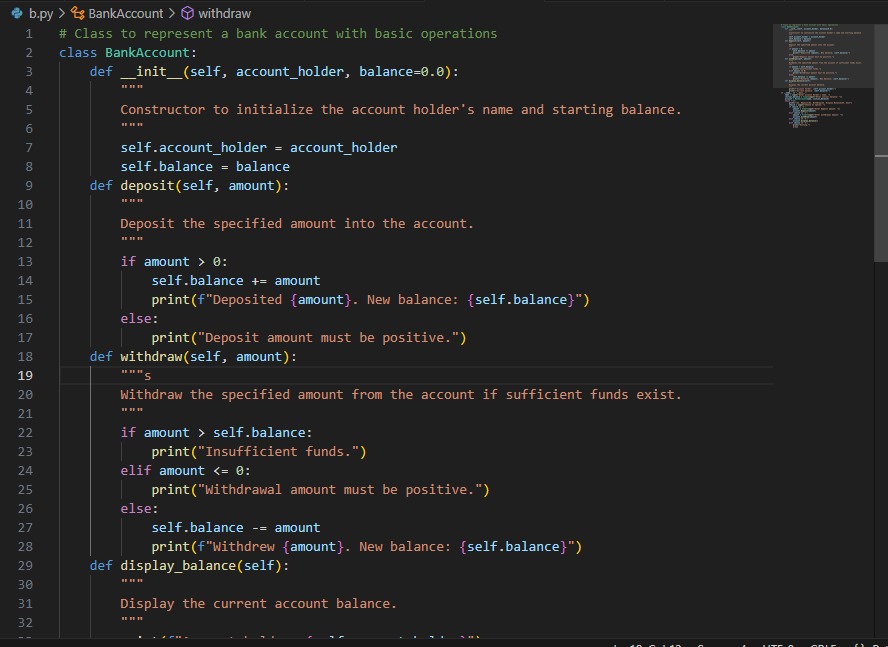
**Roll No:** 2503A51L07

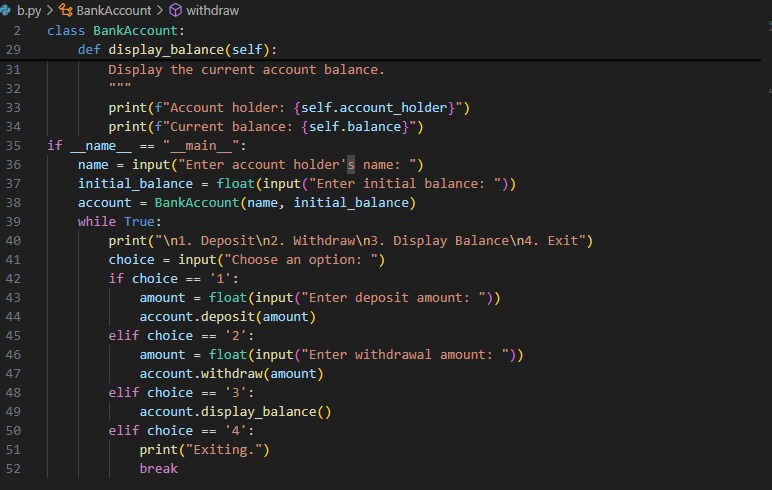
**Name:** Bathini Sahasra

**Batch:** 24BTCAICSB19

**Task #1:**

**Prompt:** Write a class definition comment and start the constructor for a class called BankAccount with account\_holder and balance attributes. Auto-complete the rest of the class, including methods to deposit, withdraw, and display balance.And also ask for the inputs from the users.

**Code Generated:  Output:**

****

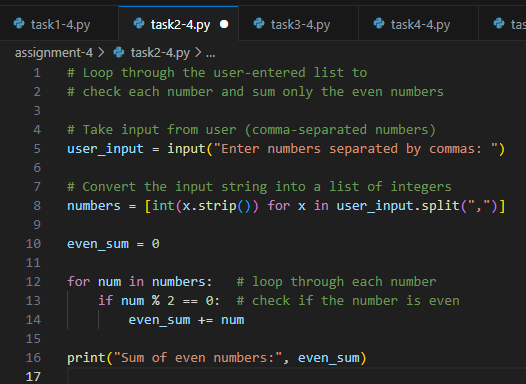
**OBSERVATION:**

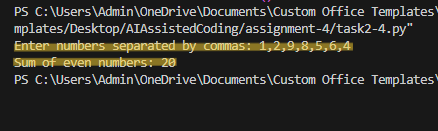
In this task, Copilot generated the complete BankAccount class after I gave the initial prompt. It provided a clear constructor and methods like deposit, withdraw, and display balance. I observed how object-oriented programming can be structured with minimal input fromme. This showed me how Copilot reduces effort while still producing correct and interactive code.

**Task #2:**

**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. Ask for the list from the user.

**Code Generated:**





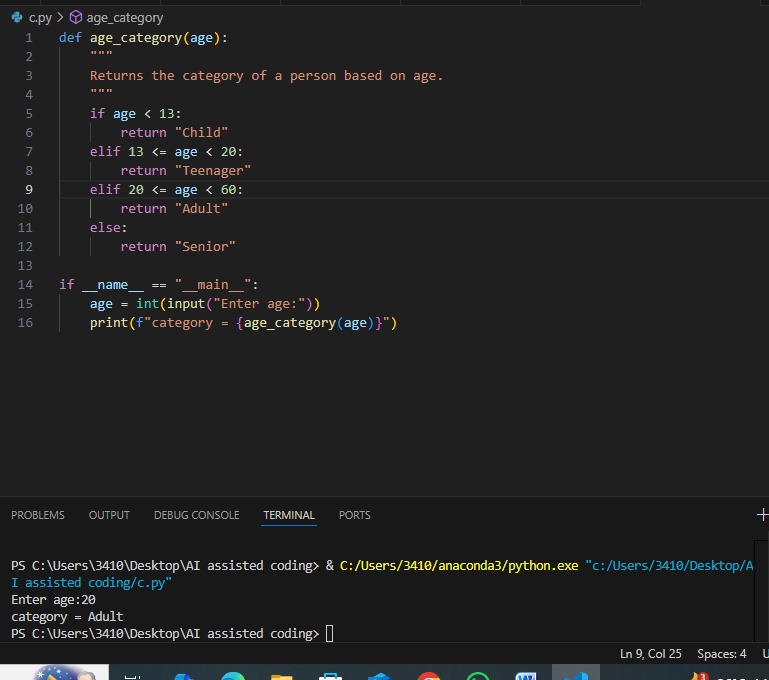
**Observation:**

This task demonstrated how to use a for loop and conditional statements to work with lists. Copilot completed the logic for checking even numbers and summing them efficiently. Taking input from the user made the program flexible for different lists. I understood how iteration and conditionals can be combined to solve real problems.

**Task #3:**

**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. Complete the conditionals and user gives the input.

**Code Generated:**

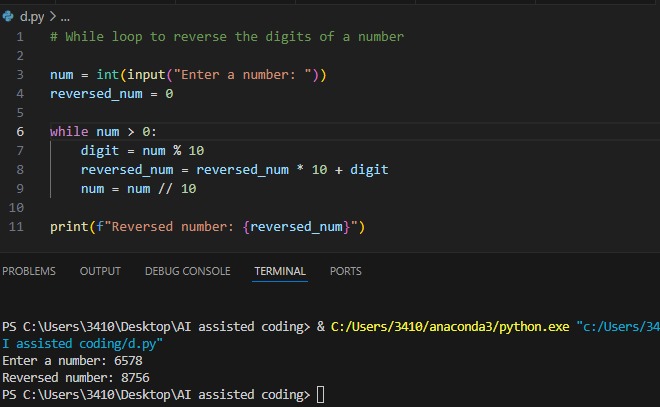
****

**Output:**

**Observation:**

Copilot completed the conditional structure for classifying age groups into child, teenager, adult, or senior. I noticed how it quickly generated accurate if-elif-else blocks from just the function start. The output confirmed that the logic worked for all ranges of ages. This task showed how Copilot saves time by providing reliable conditional logic automatically.

**Task #4:** Write a comment and start a while loop to reverse the digits of a number. Complete the loop logic and ask for the number from the user.

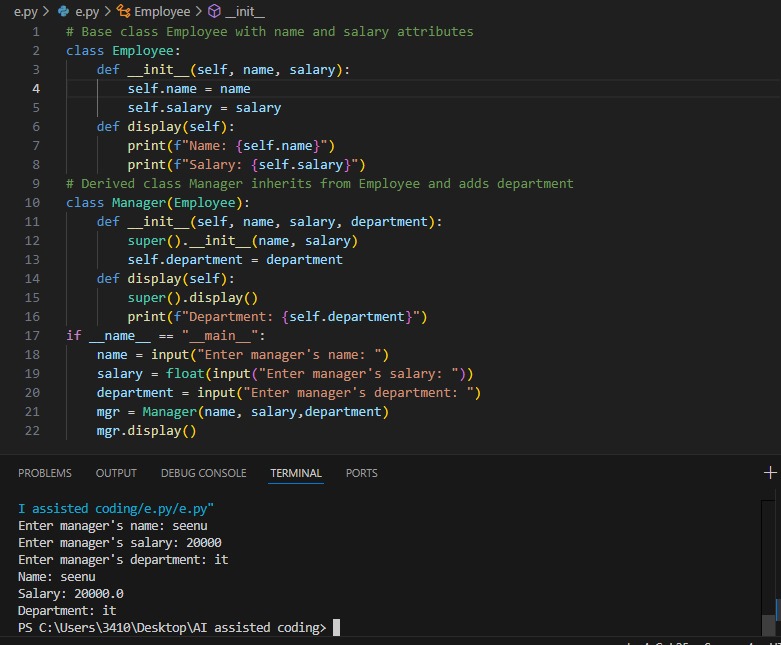
**Code Generated:**

**Output:**

**Observation:**

In this task, Copilot provided the logic to reverse digits using a while loop with modulus and integer division. I only began with a comment and loop, and it handled the rest. Running the code gave the expected reversed number. I observed how Copilot can easily generate solutions for number-based problems that usually take time to think through.

**Task #5:**

**Prompt:** Begin a class Employee with attributes name and salary. Then, start a derived class Manager that inherits from Employee and adds a department. Complete the methods and constructor chaining.

**Code Generated:**

**Output:**

**Observation:**

Copilot generated both the base Employee class and the derived Manager class with constructor chaining and department handling. I saw how inheritance and method overriding were applied correctly without me writing detailed code. The output clearly demonstrated object-oriented principles. This task made me realize how Copilot can simplify learning complex concepts like inheritance in Python.