**Name: Bollam Sathvika**

**HTNO:2403A51344**

**Batchno:14**

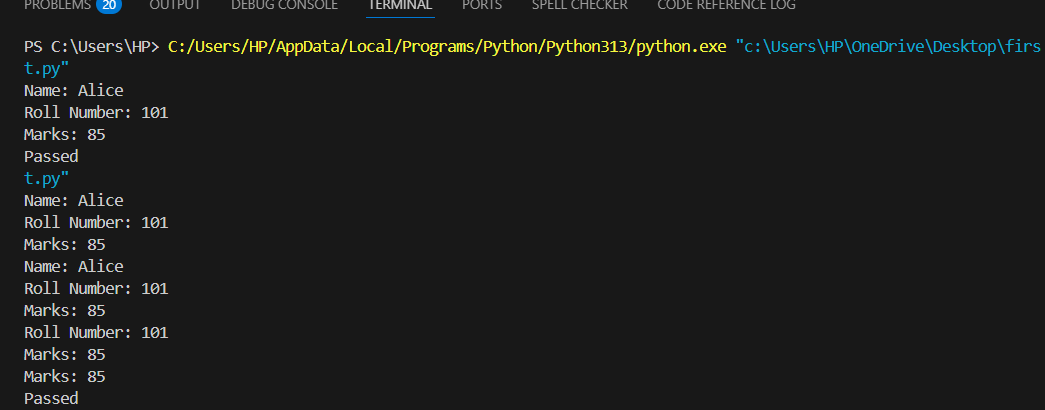
**Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals**

**Task 1:**

**Prompt:**

Create a Python class named Student with attributes name, roll\_number, and marks, and add methods display\_details() to print the student's information and is\_passed() to check if marks are greater than or equal to 40 and print 'Passed' or 'Failed'.

**Code:  
**

**Output:  
**

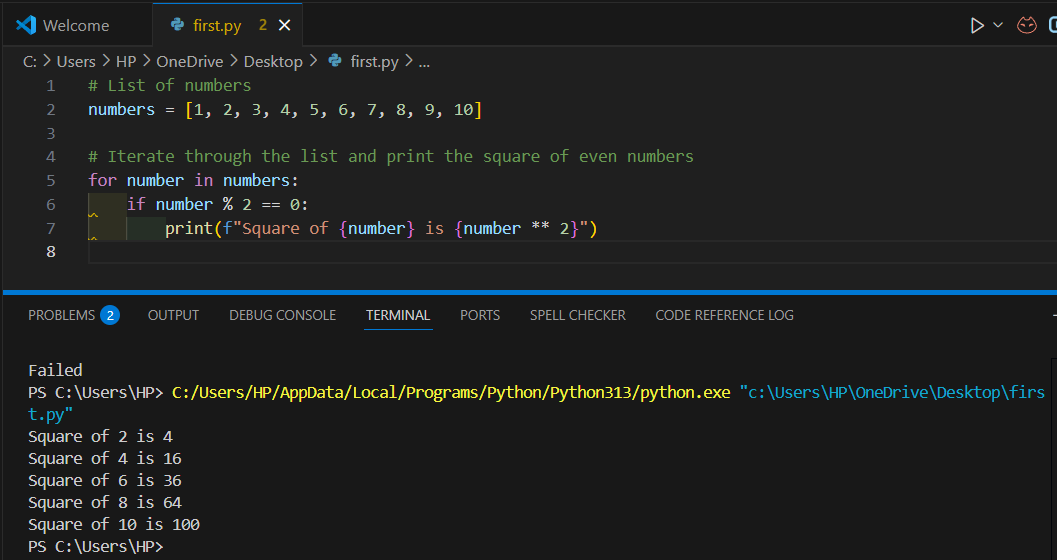
**Observation:**

The Python class Student was successfully created with attributes name, roll\_number, and marks. The method display\_details() correctly prints the student's information in a structured format. The method is\_passed() accurately checks if the marks are greater than or equal to 40 and prints "Passed" or "Failed" accordingly. The class works as expected when multiple student instances are created with different marks, demonstrating the use of if-else conditions effectively.

**Task2:**

**Prompt:** Write a for loop to iterate through a list of numbers, and calculate and print the square of even numbers only using if number % 2 == 0 condition

**Code and Output:**

****

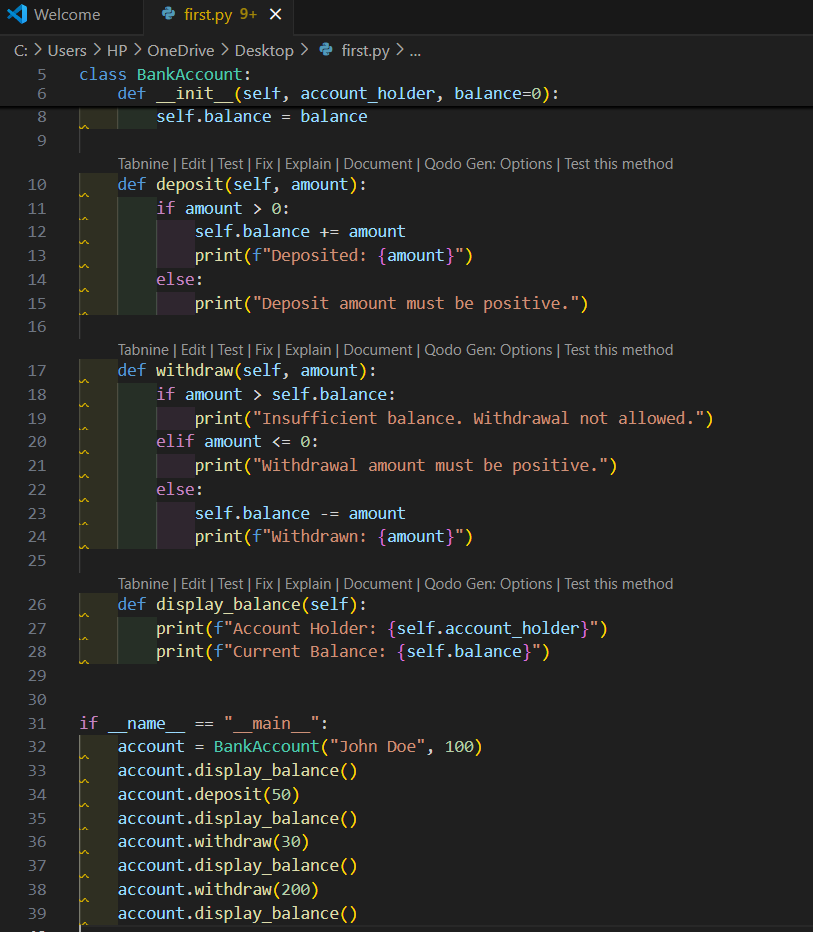
**Observation:**

The for loop successfully iterates through the list of numbers. Copilot correctly suggests the conditional check if number % 2 == 0 to identify even numbers and calculates their square using number \*\* 2, then prints the result. The final output shows the squares of only the even numbers in the list.

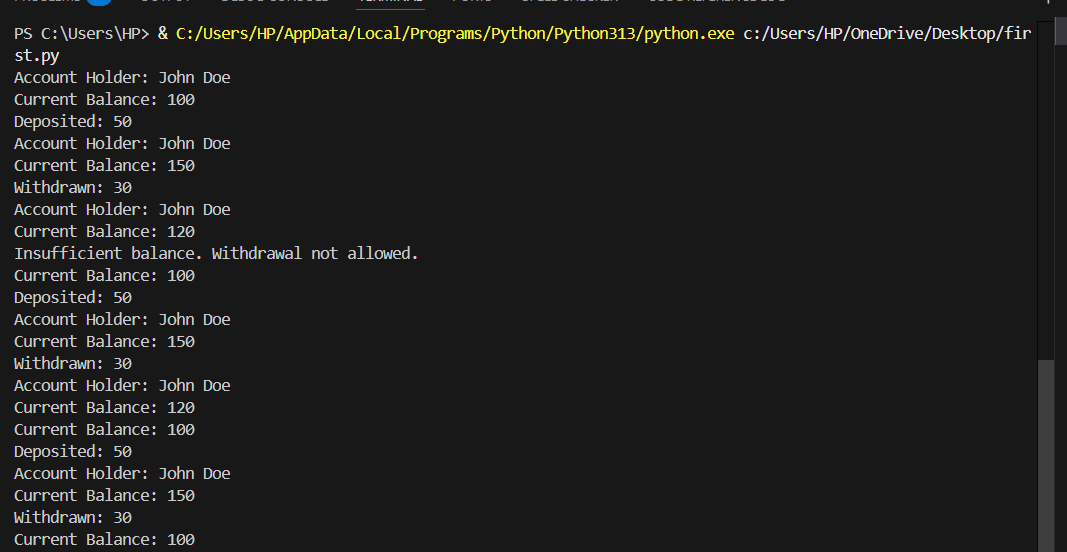
**Task3:**

**Prompt:** Create a Python class called BankAccount with attributes account\_holder and balance. Add methods deposit() to add money to the balance, withdraw() to subtract money only if sufficient balance is available, and display\_balance() to show the current balance. Ensure that withdrawing more than the available balance is not allowed

**Code:**

****

**Output:**

****

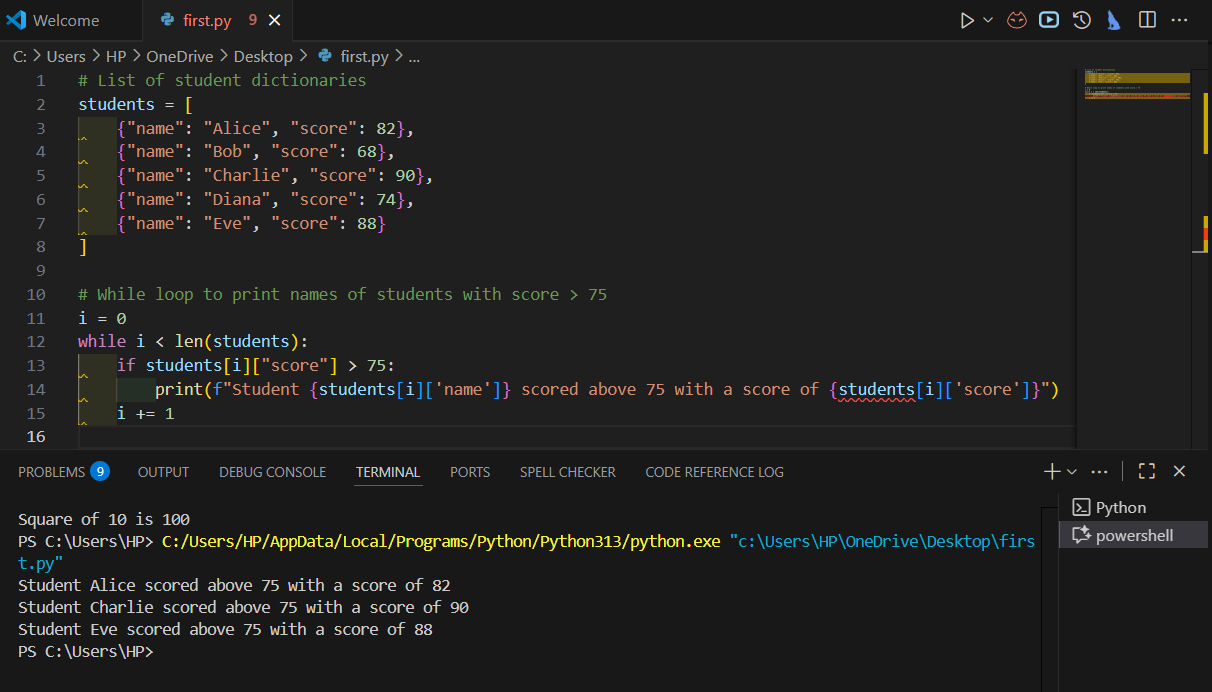
**Observation:**

The BankAccount class was successfully created with attributes account\_holder and balance. The deposit() method correctly increases the balance by the specified amount. The withdraw() method uses an if condition to check if the balance is sufficient before subtracting the withdrawal amount, preventing overdrawing. The display\_balance() method correctly shows the current balance. The class works as expected, handling deposits, withdrawals, and insufficient balance situations properly.

**Task4:**

**Prompt:** Define a list of student dictionaries with keys name and score. Write a while loop to iterate through the list and print the names of students whose score is greater than 75, using proper condition checks and formatted output

**Code and Output:**

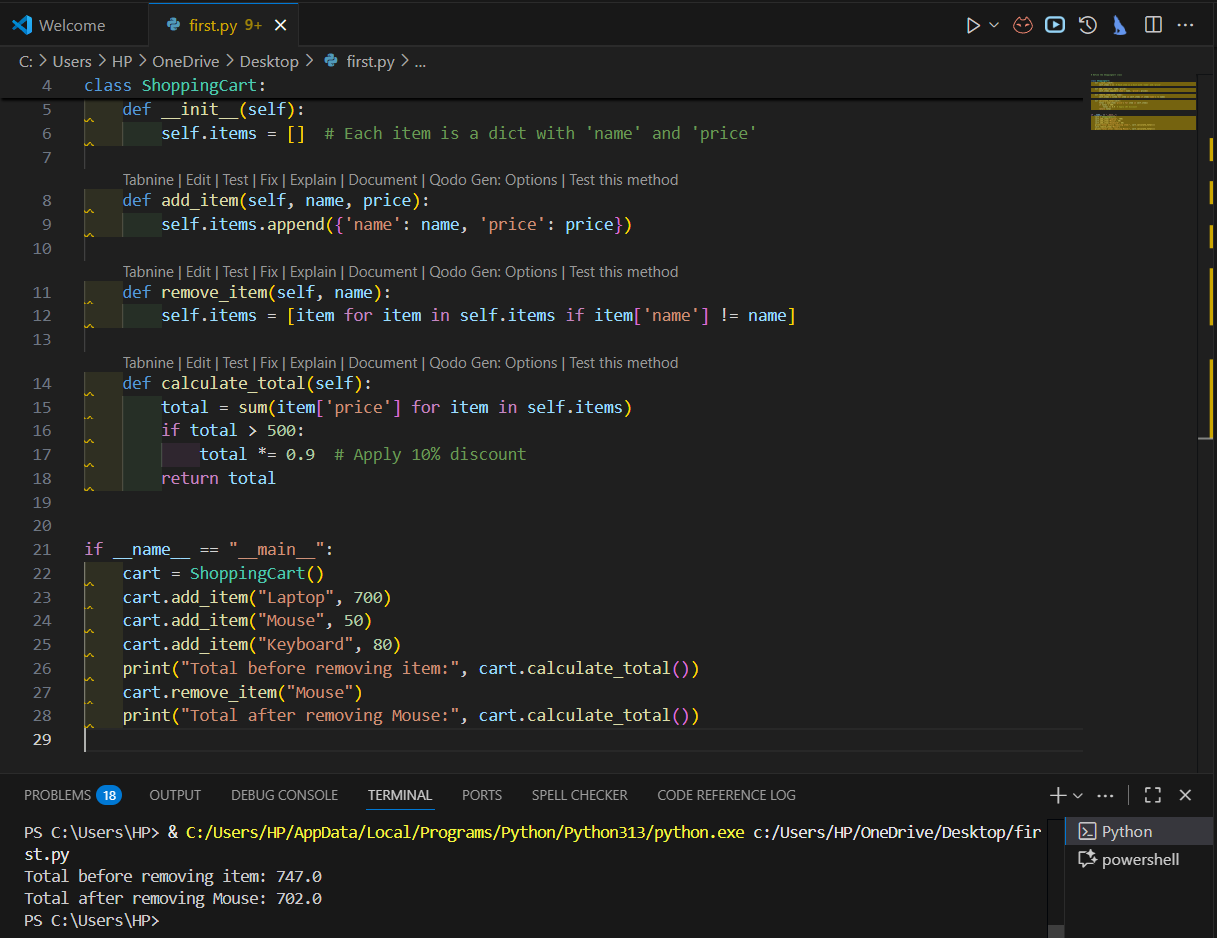
****

**Observation:**

The list of student dictionaries was successfully created, each containing name and score keys. Copilot generated a functional while loop that iterates through the list using an index. It correctly checks if a student’s score is greater than 75 using an if condition, and prints the student’s name in a formatted way. The loop increments the index properly to avoid an infinite loop, and the output shows only the names of students who scored more than 75 as expected

**Task5:**

**Prompt:** Create a Python class called ShoppingCart with an empty list items. Generate methods add\_item() to add an item with its price, remove\_item() to remove an item by name, and calculate\_total() to loop through items, apply a discount (e.g., 10% off if total exceeds 500), and return the total bill.

**Code and Output:  
**

**Observation:**

The ShoppingCart class was successfully created with an empty list attribute items. Copilot generated the add\_item() method to add items as dictionaries containing name and price, and the remove\_item() method to remove an item by matching its name. The calculate\_total() method uses a loop to sum item prices, and an if-else statement applies a discount when the total exceeds a certain threshold (e.g., 10% off if total > 500). The class works as expected, managing items and calculating the total bill with correct discount logic.