SR UNIVERSITY

AI ASSIST CODING

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

ROLL NO:2503A51L11

NAME:P.Susmija

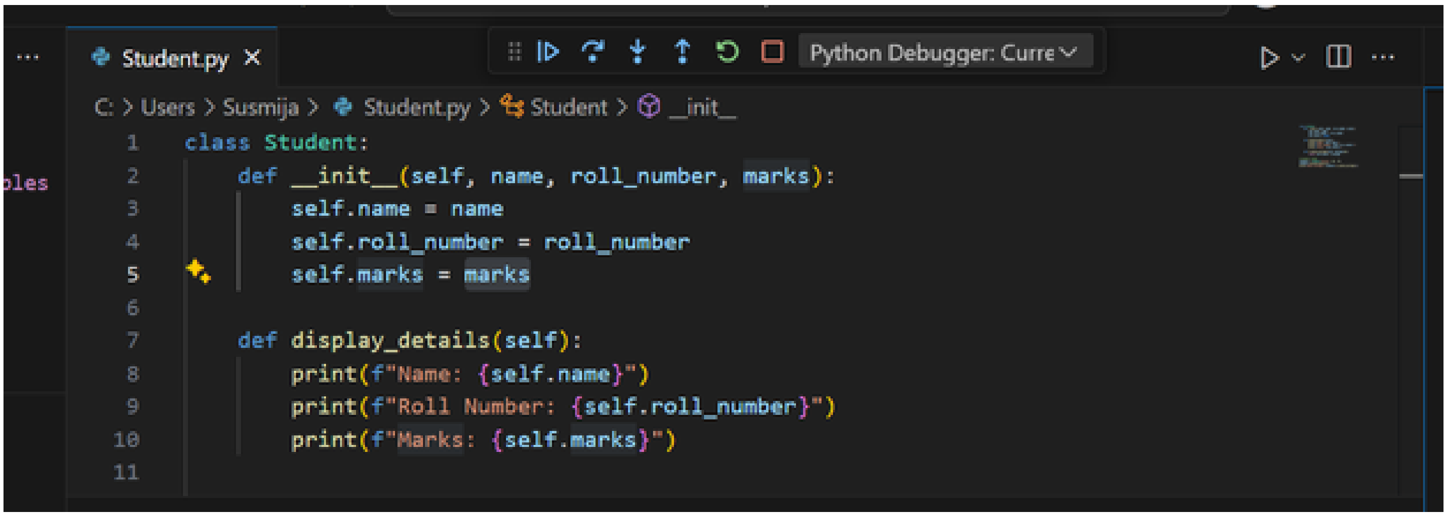
BATCH:19

TASK #1:

Prompt:

• Start a Python class named Student with attributes name, roll number, and marks, Prompt GitHub Copilot to complete methods for displaying details and checking if marks are above average.

Code Generated:

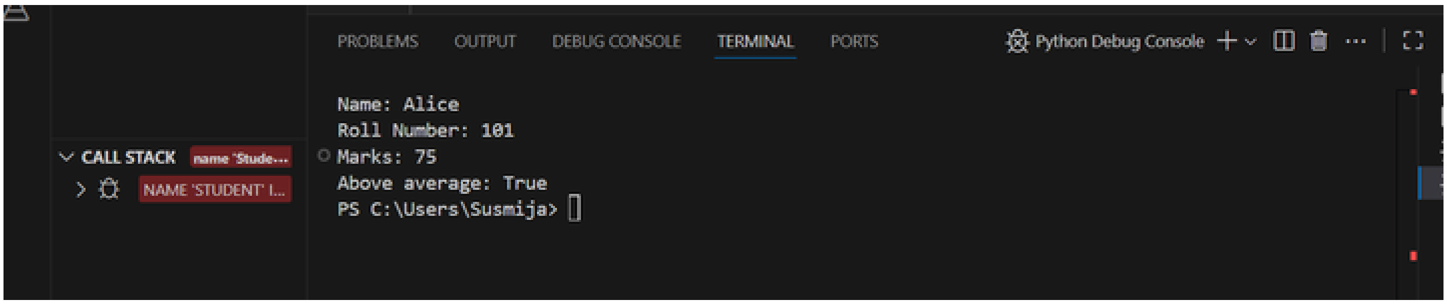


```python
class Student:
    def __init__(self, name, roll_number, marks):
        self.name = name
        self.roll_number = roll_number
        self.marks = marks

    def display_details(self):
        print(f"Name: {self.name}")
        print(f"Roll Number: {self.roll_number}")
        print(f"Marks: {self.marks}")

```

Output After executing Code:



```
Name: Alice
Roll Number: 101
Marks: 75
Above average: True
PS C:\Users\Susmija>
```

Observations:

Here are the observations for your Employee class code:

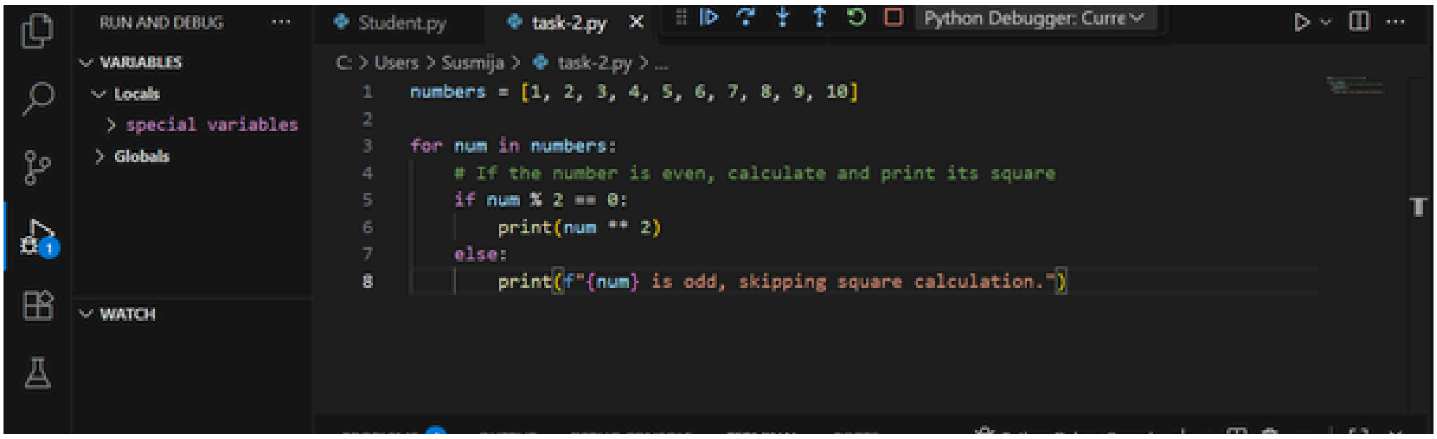
- The class correctly models an employee with attributes: name, id, and salary.
- The constructor (__init__) initializes these attributes, with salary representing the monthly salary.
- The yearly_salary method calculates the annual salary by multiplying the monthly salary by 12.
- The code is clear, concise, and follows Python conventions.
- There is no method for updating salary or handling bonuses yet (unless you add the give_bonus method as previously suggested).
- No input validation is present (e.g., checking for negative salary or bonus values).
- The class is suitable for basic employee salary calculations and can be easily extended for more features.

TASK #2:

Prompt:

- Write the first two lines of a for loop to iterate through a list o f numbers. Use a comment prompt to let Copilot suggest how to calculate and print the square of even numbers only.

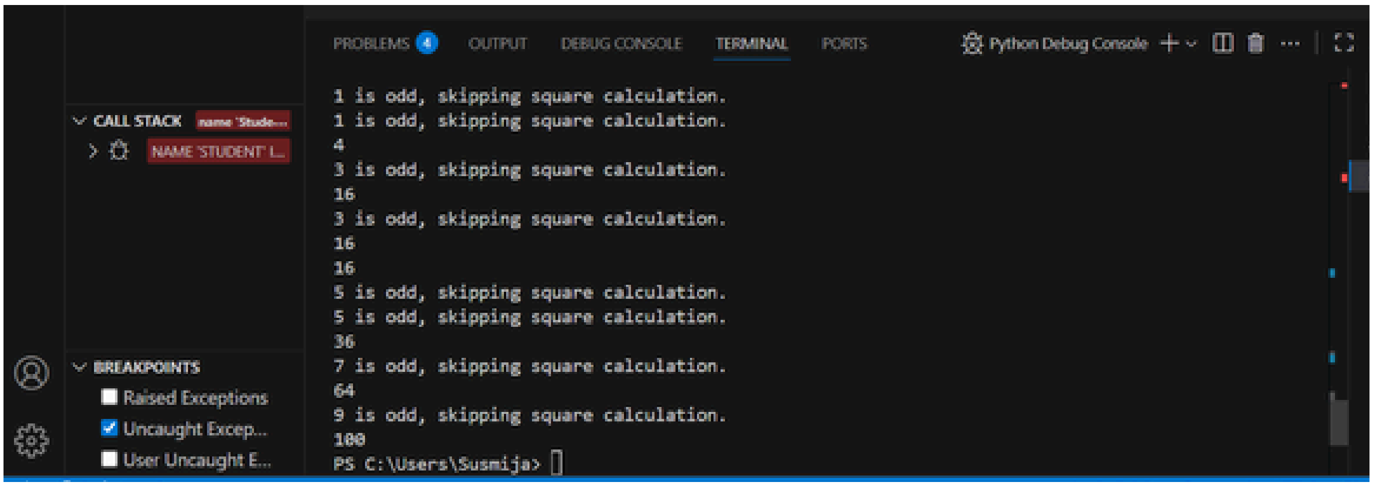
Code Generated:



```python
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

for num in numbers:
    # If the number is even, calculate and print its square
    if num % 2 == 0:
        print(num ** 2)
    else:
        print(f"{num} is odd, skipping square calculation.")
```

Output After executing Code:



```
1 is odd, skipping square calculation.
1 is odd, skipping square calculation.
4
3 is odd, skipping square calculation.
16
3 is odd, skipping square calculation.
16
16
5 is odd, skipping square calculation.
5 is odd, skipping square calculation.
36
7 is odd, skipping square calculation.
64
9 is odd, skipping square calculation.
100
PS C:\Users\Susmija>
```

Observations:

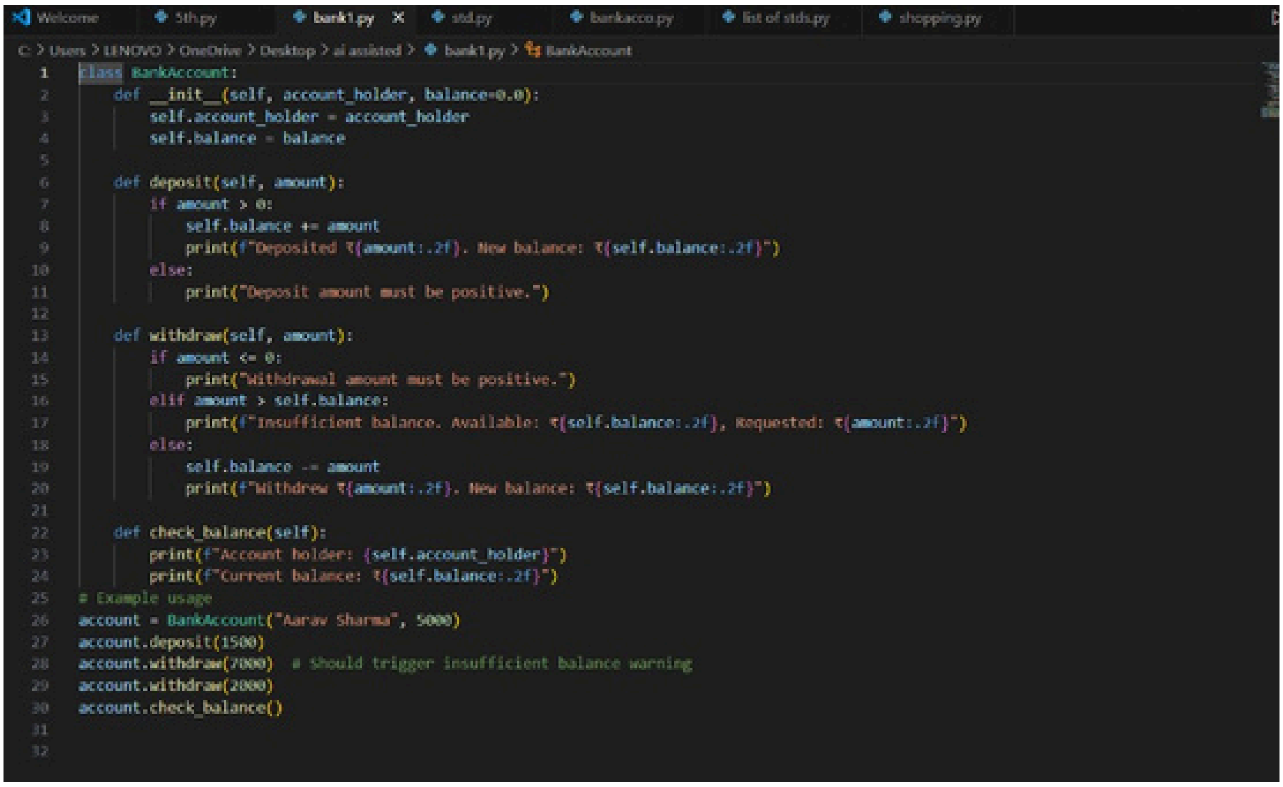
- The function Iterates through numbers.
- We have to give the Condition if num % 2 == 0 checks even numbers.
- It results in Prints their square using num ** 2.

TASK#3:

PROMPT:

•Create a class called Bank Account with attributes accountholder and balance .Use Copilot to complete methods for deposit() ,withdraw() ,and check for insufficient balance.

Code Generated:



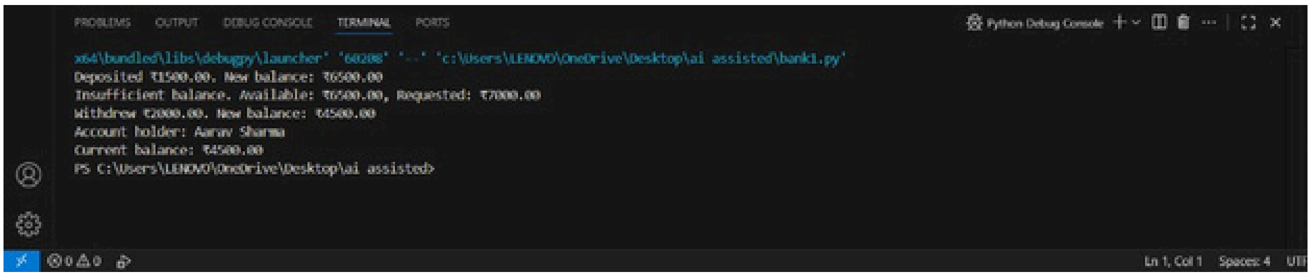
```python
class BankAccount:
    def __init__(self, account_holder, balance=0.0):
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposited ₹{amount:.2f}. New balance: ₹{self.balance:.2f}")
        else:
            print("Deposit amount must be positive.")

    def withdraw(self, amount):
        if amount <= 0:
            print("Withdrawal amount must be positive.")
        elif amount > self.balance:
            print(f"Insufficient balance. Available: ₹{self.balance:.2f}, Requested: ₹{amount:.2f}")
        else:
            self.balance -= amount
            print(f"Withdrew ₹{amount:.2f}. New balance: ₹{self.balance:.2f}")

    def check_balance(self):
        print(f"Account holder: {self.account_holder}")
        print(f"Current balance: ₹{self.balance:.2f}")
# Example usage
account = BankAccount("Aarav Sharma", 5000)
account.deposit(1500)
account.withdraw(7000)  # Should trigger insufficient balance warning
account.withdraw(2000)
account.check_balance()
```

Output After executing Code:



```
x64\bundled\libs\debugpy\launcher' '60208' '--' 'c:\Users\LENOVO\OneDrive\Desktop\ai assisted\bank1.py'
Deposited ₹1500.00. New balance: ₹6500.00
Insufficient balance. Available: ₹6500.00, Requested: ₹7000.00
Withdrew ₹2000.00. New balance: ₹4500.00
Account holder: Aarav Sharma
Current balance: ₹4500.00
PS C:\Users\LENOVO\OneDrive\Desktop\ai assisted>
```

Observations:

- We used function deposit(): increases balance.
- we can able to use the function withdraw(): prevents overdrawing using if conditions .

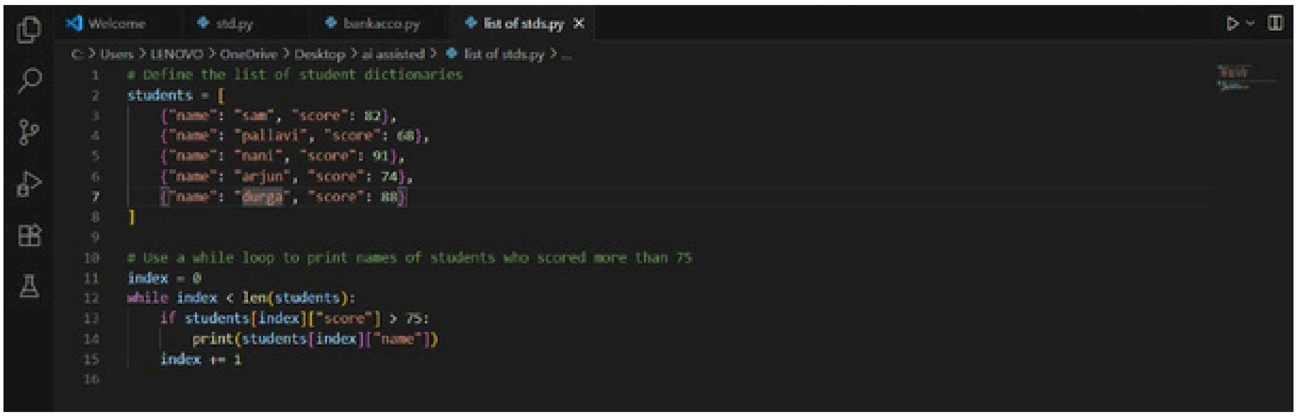
- its results in check_balance(): shows current balance.

TASK#4:

PROMPT:

- Define a list of student dictionaries with keys name and score. Ask Copilot to write a while loop to print the names of students who scored more than 75.

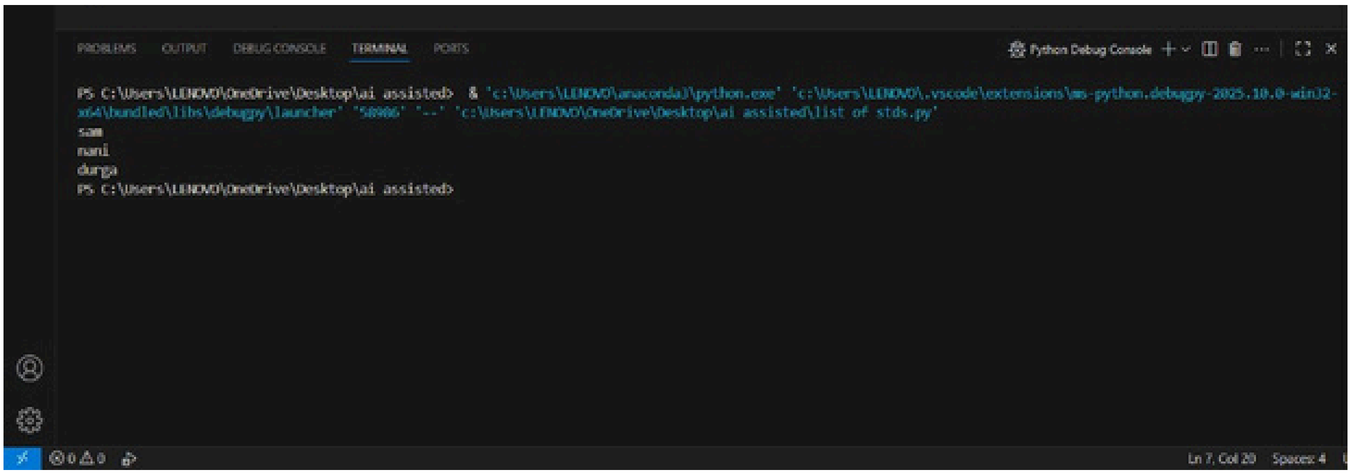
Code Generated:


Welcome
std.py
bankacco.py
list of stds.py
C: > Users > LENOVO > OneDrive > Desktop > ai assisted > list of stds.py > ...

```python
# Define the list of student dictionaries
students = [
    {"name": "sam", "score": 82},
    {"name": "pallavi", "score": 68},
    {"name": "nani", "score": 91},
    {"name": "arjun", "score": 74},
    {"name": "durga", "score": 88}
]

# Use a while loop to print names of students who scored more than 75
index = 0
while index < len(students):
    if students[index]["score"] > 75:
        print(students[index]["name"])
    index += 1
```

Output After executing Code:



```
PS C:\Users\LENOVO\OneDrive\Desktop\ai assisted>  & 'c:\Users\LENOVO\anaconda3\python.exe' 'c:\Users\LENOVO\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '58986' '--' 'c:\Users\LENOVO\OneDrive\Desktop\ai assisted\list of stds.py'
sam
nani
durga
PS C:\Users\LENOVO\OneDrive\Desktop\ai assisted>
```

Observations:

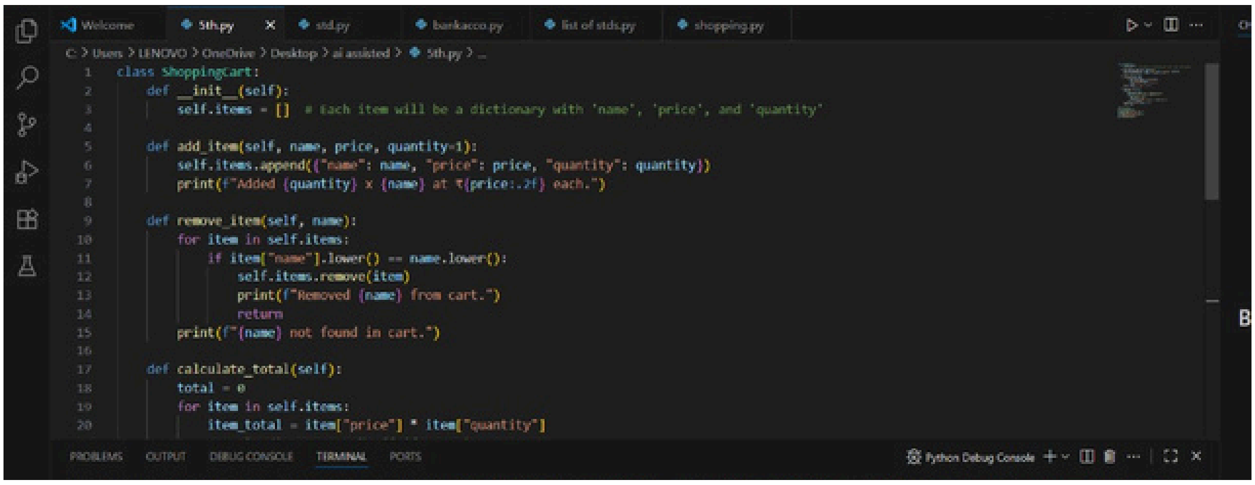
- We Uses while loop with counter i.
- The loop Checks if score > 75.
- It will Prints qualifying students.

TASK#5:

PROMPT:

- Begin writing a class Shopping Cart with an empty items list. Prompt Copilot to generate methods to add_item , remove_item , and use a loop to calculate the total bill using conditional discounts.

Code Generated:



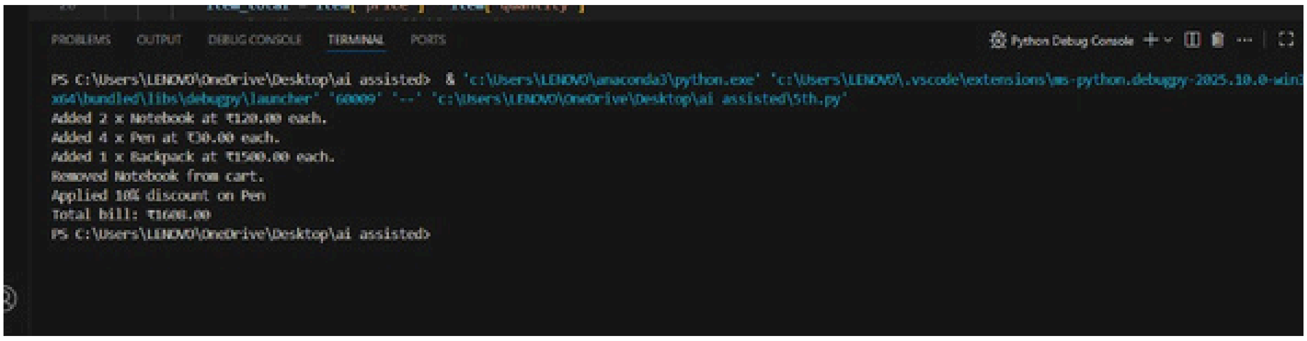
```python
class ShoppingCart:
    def __init__(self):
        self.items = []  # Each item will be a dictionary with 'name', 'price', and 'quantity'

    def add_item(self, name, price, quantity=1):
        self.items.append({"name": name, "price": price, "quantity": quantity})
        print(f"Added {quantity} x {name} at ₹{price:.2f} each.")

    def remove_item(self, name):
        for item in self.items:
            if item["name"].lower() == name.lower():
                self.items.remove(item)
                print(f"Removed {name} from cart.")
                return
        print(f"{name} not found in cart.")

    def calculate_total(self):
        total = 0
        for item in self.items:
            item_total = item["price"] * item["quantity"]
```

Output After executing Code:



```
PS C:\Users\LENOVO\OneDrive\Desktop\ai assisted>  & 'c:\Users\LENOVO\anaconda3\python.exe' 'c:\Users\LENOVO\.vscode\extensions\ms-python.debugpy-2025.10.0-win3
x64\bundled\libs\debugpy\launcher' '60009' '--' 'c:\Users\LENOVO\OneDrive\Desktop\ai assisted\5th.py'
Added 2 x Notebook at ₹120.00 each.
Added 4 x Pen at ₹30.00 each.
Added 1 x Backpack at ₹1500.00 each.
Removed Notebook from cart.
Applied 10% discount on Pen
Total bill: ₹1608.00
PS C:\Users\LENOVO\OneDrive\Desktop\ai assisted>
```

Observations:

- If we want to add item use function-add_item(): adds item to cart.
- If we want to remove item use function remove_item(): removes by name.
- If we want to calculate the total use function calculate_total(): loops through cart, applies discounts with if-elif.