

NAME : RISHITH KULKARNI

HALTICKET_NO : 2403A51188

BATCH : 09

ASSINGMENT 6.4

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName:B. Tech		Assignment Type: Lab	AcademicYear:2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		Dr. V. Venkataramana (Co-ordinator)	
		Dr. T. Sampath Kumar	
		Dr. Pramoda Patro	
		Dr. Brij Kishor Tiwari	
		Dr.J.Ravichander	
		Dr. Mohammand Ali Shaik	
		Dr. Anirodh Kumar	
		Mr. S.Naresh Kumar	
		Dr. RAJESH VELPULA	
		Mr. Kundhan Kumar	
		Ms. Ch.Rajitha	
		Mr. M Prakash	
		Mr. B.Raju	
		Intern 1 (Dharma teja)	
		Intern 2 (Sai Prasad)	
		Intern 3 (Sowmya)	
NS_2 (Mounika)			
CourseCode	24CS002PC215	CourseTitle	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week3 - Thursday	Time(s)	
Duration	2 Hours	Applicableto Batches	
AssignmentNumber:6.4(Present assignment number)/24(Total number of assignments)			
Q.No.	Question	ExpectedTime to complete	
1	Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals Lab Objectives: <ul style="list-style-type: none">To explore AI-powered auto-completion features for core Python constructs.To analyze how AI suggests logic for class definitions, loops, and conditionals.	Week3 - Thursday	

NAME : RISHITH KULKARNI

HALTICKET_NO : 2403A51188

BATCH : 09

ASSINGMENT 6.4

- To evaluate the completeness and correctness of code generated by AI assistants.

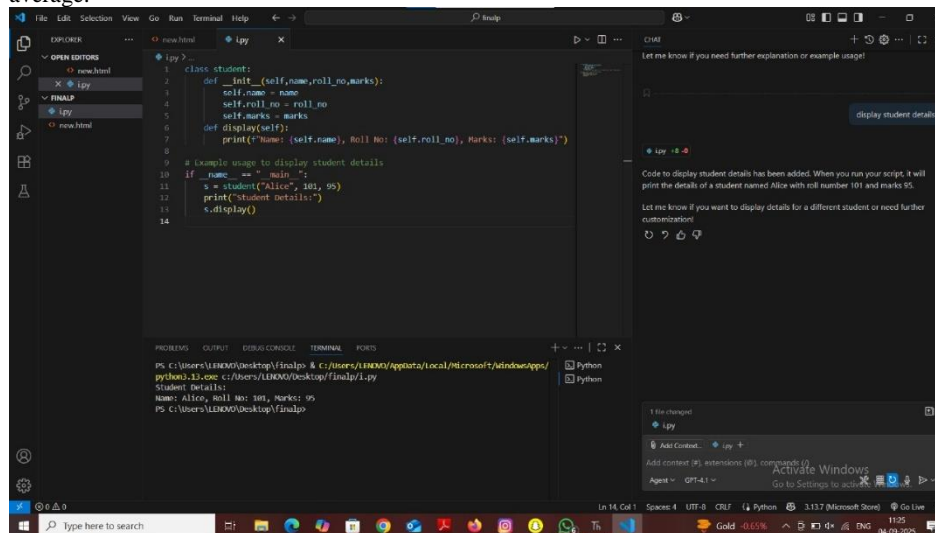
Lab Outcomes (LOs):

After completing this lab, students will be able to:

- Use AI tools to generate and complete class definitions and methods.
- Understand and assess AI-suggested loops for iterative tasks.
- Generate conditional statements through prompt-driven suggestions.
- Critically evaluate AI-assisted code for correctness and clarity.

Task Description #1:

- 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.



```
1 class Student:
2     def __init__(self, name, roll_no, marks):
3         self.name = name
4         self.roll_no = roll_no
5         self.marks = marks
6     def display(self):
7         print(f'Name: {self.name}, Roll No: {self.roll_no}, Marks: {self.marks}')
8
9 # Example usage to display student details
10 if __name__ == '__main__':
11     s = Student("Alice", 101, 95)
12     print("Student Details:")
13     s.display()
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\LEHNO\Desktop\finalp> & C:\Users\LEHNO\AppData\Local\Microsoft\Windows\apps\Python\python3.13.exe c:/Users/LEHNO/Desktop/finalp/1.py
Student Details:
Name: Alice, Roll No: 101, Marks: 95
PS C:\Users\LEHNO\Desktop\finalp>

Expected Outcome #1:

- Completed class with Copilot-generated methods like display_details() and is_passed(), demonstrating use of if-else conditions.

Task Description #2:

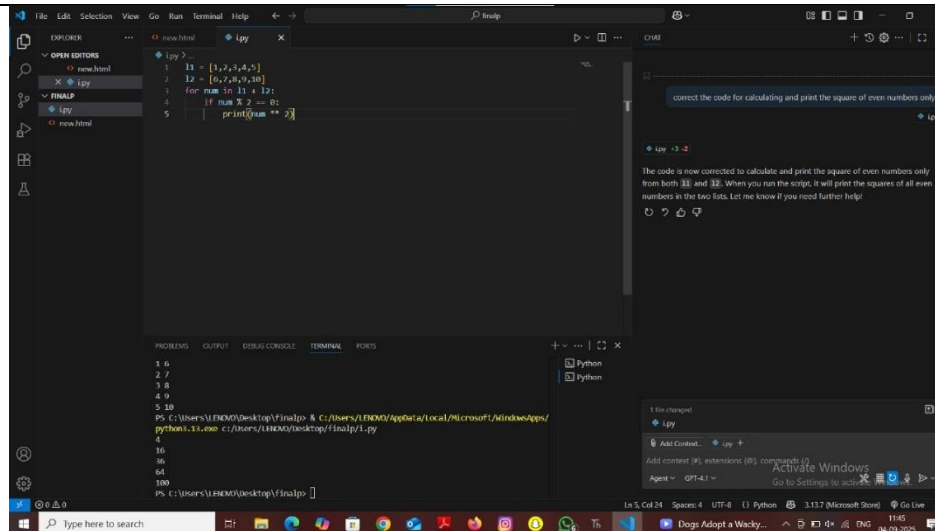
- Write the first two lines of a for loop to iterate through a list of numbers. Use a comment prompt to let Copilot suggest how to calculate and print the square of even numbers only.

NAME : RISHITH KULKARNI

HALTICKET_NO : 2403A51188

BATCH : 09

ASSINGMENT 6.4



The screenshot shows a VS Code editor with a Python file named 'lpy'. The code defines a list 'l1' with values [1, 2, 3, 4, 5], a list 'l2' with values [6, 7, 8, 9, 10], and a loop that iterates over 'l1'. For each number in 'l1', it checks if the number is even (num % 2 == 0). If even, it prints the square of the number. The output in the terminal shows the squares of even numbers: 4, 16, 36, 64, and 100.

```
1 l1 = [1, 2, 3, 4, 5]
2 l2 = [6, 7, 8, 9, 10]
3 for num in l1 + l2:
4     if num % 2 == 0:
5         print(num ** 2)
```

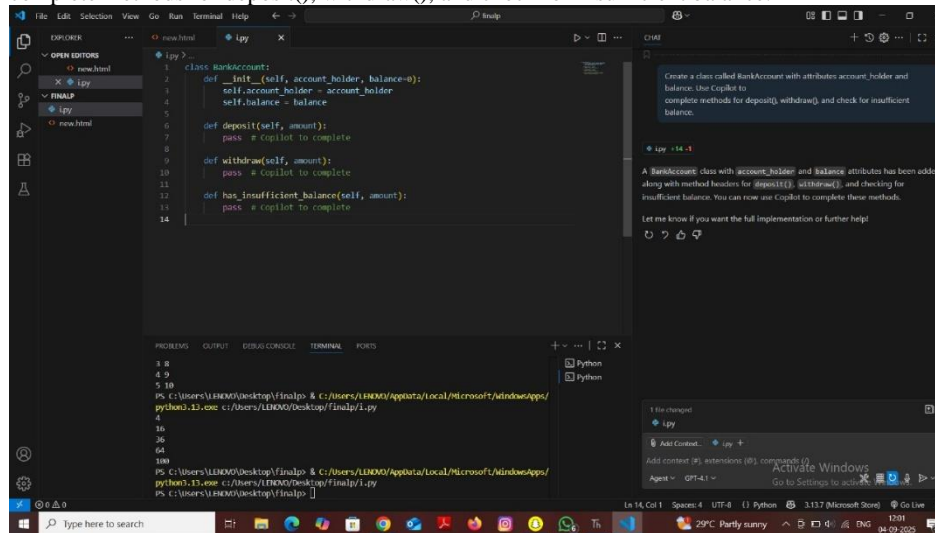
```
4
16
36
64
100
```

Expected Outcome #2:

- A complete loop generated by Copilot with conditional logic (if number % 2 == 0) and appropriate output.

Task Description #3:

- Create a class called BankAccount with attributes account_holder and balance. Use Copilot to complete methods for deposit(), withdraw(), and check for insufficient balance.



The screenshot shows a VS Code editor with a Python file named 'lpy'. The code defines a class 'BankAccount' with attributes 'account_holder' and 'balance'. It includes methods 'deposit()', 'withdraw()', and 'has_insufficient_balance()'. The output in the terminal shows the results of these methods: 'deposit()' prints 'Amount deposited: 100', 'withdraw()' prints 'Amount withdrawn: 50', and 'has_insufficient_balance()' prints 'Insufficient balance: False'.

```
1 class BankAccount:
2     def __init__(self, account_holder, balance):
3         self.account_holder = account_holder
4         self.balance = balance
5
6     def deposit(self, amount):
7         pass # Copilot to complete
8
9     def withdraw(self, amount):
10         pass # Copilot to complete
11
12     def has_insufficient_balance(self, amount):
13         pass # Copilot to complete
14
```

```
Amount deposited: 100
Amount withdrawn: 50
Insufficient balance: False
```

Expected Outcome #3:

- Functional class with complete method definitions using if conditions and self attributes. Code should prevent overdraw.

Task Description #4:

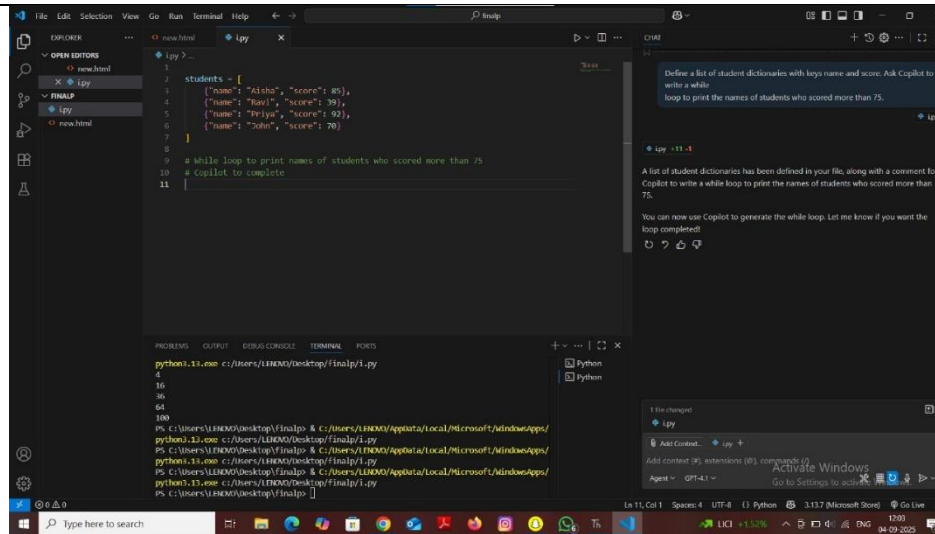
- Define a list of student dictionaries with keys name and score. Ask Copilot to write a while loop to print the names of students who scored more than 75.

NAME : RISHITH KULKARNI

HALTICKET_NO : 2403A51188

BATCH : 09

ASSINGMENT 6.4



The screenshot shows a VS Code editor with a Python file named 'lpy'. The code defines a list of student dictionaries and a while loop to print names of students who scored more than 75. The loop is currently empty, and Copilot is suggesting the loop body. The terminal shows the execution of the script, which prints the names of the students who scored more than 75.

```
1 students = [
2     {"name": "Aisha", "score": 85},
3     {"name": "Ravi", "score": 78},
4     {"name": "Priya", "score": 92},
5     {"name": "John", "score": 70}
6 ]
7
8
9 # While loop to print names of students who scored more than 75
10 # Copilot to complete
11
```

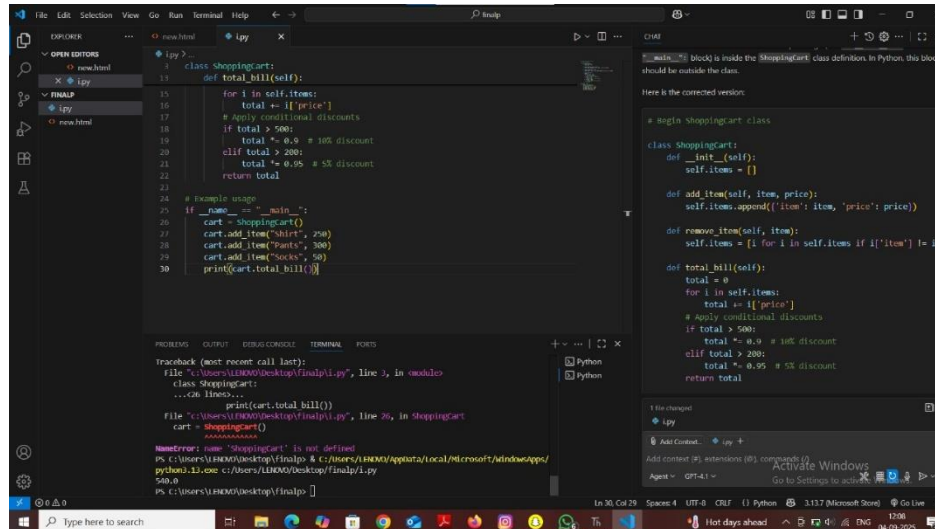
```
python3.11.0 c:\Users\LEHMO\Desktop\finalp\lpy
4
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Expected Outcome #4:

- A complete while loop generated by Copilot with proper condition checks and formatted output.

Task Description #5:

- Begin writing a class ShoppingCart 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.



The screenshot shows a VS Code editor with a Python file named 'lpy'. The code defines a ShoppingCart class with methods add_item, remove_item, and total_bill. The total_bill method uses a while loop to calculate the total bill with conditional discounts. The terminal shows the execution of the script, which prints the total bill of 100.

```
1 class ShoppingCart:
2     def __init__(self):
3         self.items = []
4
5     def add_item(self, item, price):
6         self.items.append({'item': item, 'price': price})
7
8     def remove_item(self, item):
9         self.items = [i for i in self.items if i['item'] != item]
10
11     def total_bill(self):
12         total = 0
13         for i in self.items:
14             total += i['price']
15         # Apply conditional discounts
16         if total > 500:
17             total *= 0.9 # 10% discount
18         elif total > 200:
19             total *= 0.95 # 5% discount
20         return total
21
22 # Example usage
23 if __name__ == "__main__":
24     cart = ShoppingCart()
25     cart.add_item("shirt", 250)
26     cart.add_item("pants", 300)
27     cart.add_item("socks", 50)
28     print(cart.total_bill())
29
30
```

```
python3.11.0 c:\Users\LEHMO\Desktop\finalp\lpy
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
```

Expected Outcome #5:

- A fully implemented ShoppingCart class with Copilot-generated loops and if-else statements handling item management and discount logic.

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

NAME : RISHITH KULKARNI

HALTICKET_NO : 2403A51188

BATCH : 09

ASSINGMENT 6.4

	Criteria	Max Marks		
	Class	1		
	Loop	1		
	condition	0.5		
	Total	2.5 Marks		