AI ASSISTED CODING

ASSIGNMENT-6.4

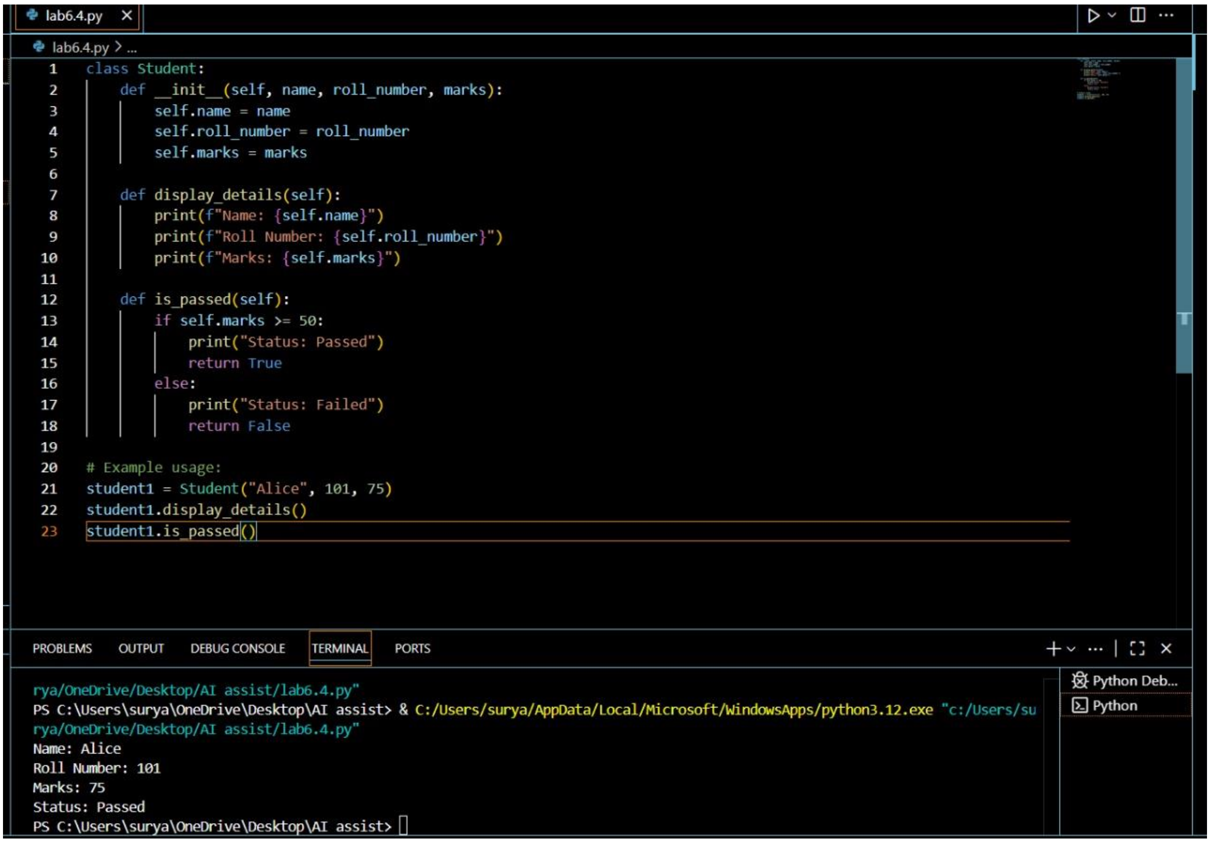
Name : R.Suryanarayana

HT.no : 2403A52038

Task 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.

Code:



```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}")

    def is_passed(self):
        if self.marks >= 50:
            print("Status: Passed")
            return True
        else:
            print("Status: Failed")
            return False

# Example usage:
student1 = Student("Alice", 101, 75)
student1.display_details()
student1.is_passed()
```

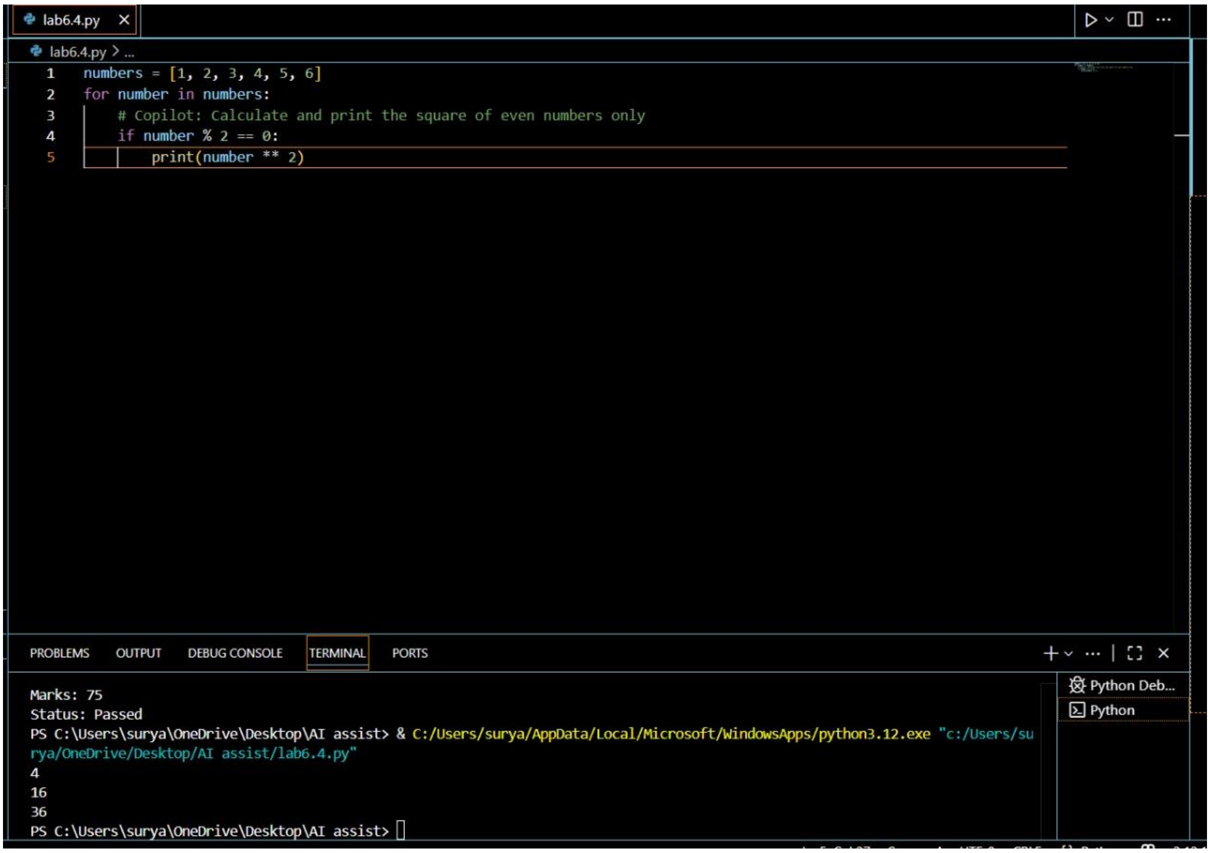
```
rya/OneDrive/Desktop/AI assist/lab6.4.py"
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/su
rya/OneDrive/Desktop/AI assist/lab6.4.py"
Name: Alice
Roll Number: 101
Marks: 75
Status: Passed
PS C:\Users\surya\OneDrive\Desktop\AI assist>
```

Observation:

Copilot-generated methods like displaying details and checking if marks are above average make the class practical and easy to use. The code is straightforward and demonstrates basic object-oriented programming and conditional checks.

Task 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.



```python
numbers = [1, 2, 3, 4, 5, 6]
for number in numbers:
    # Copilot: Calculate and print the square of even numbers only
    if number % 2 == 0:
        print(number ** 2)
```

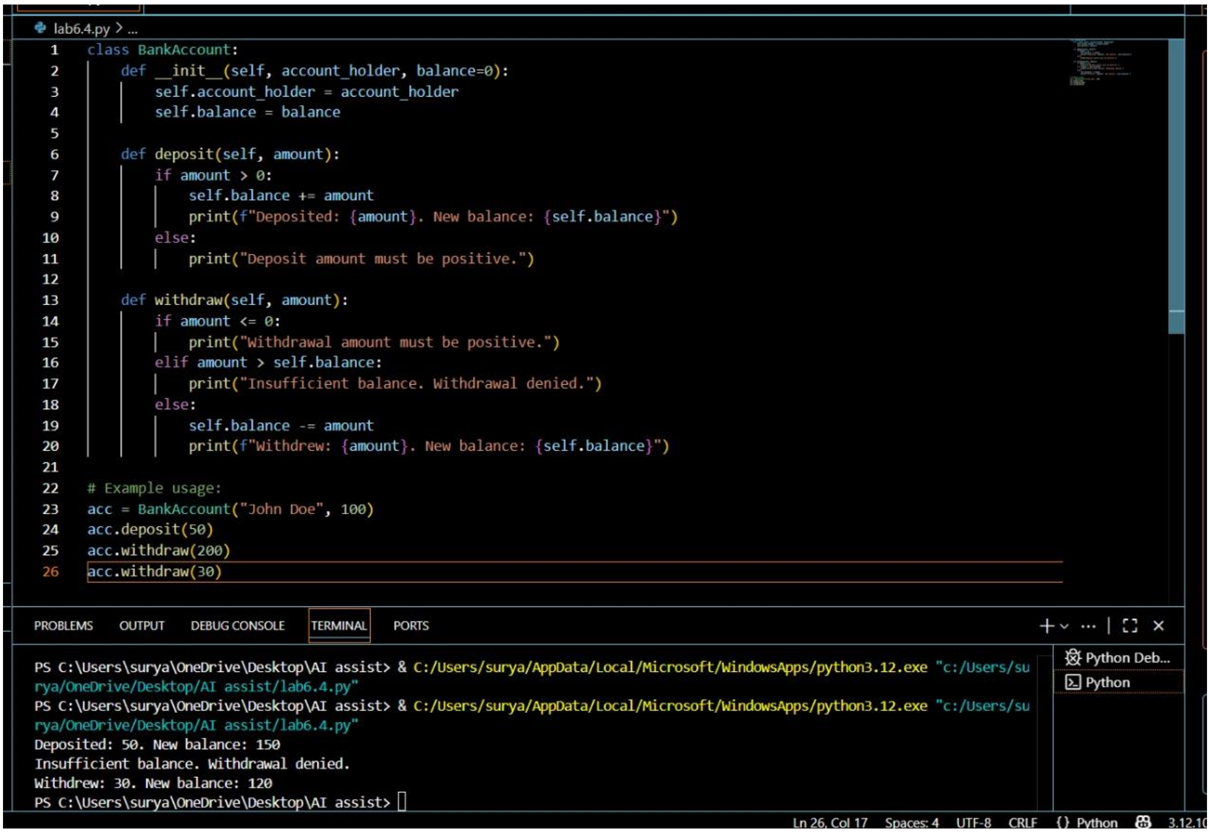
```
Marks: 75
Status: Passed
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/su
rya/OneDrive/Desktop/AI assist/lab6.4.py"
4
16
36
PS C:\Users\surya\OneDrive\Desktop\AI assist> 
```

Observation:

This task demonstrates how to use a for loop to iterate through a list of numbers and apply conditional logic to process only even numbers. By including a comment prompt, Copilot can suggest code to calculate and print the square of even numbers, making the code concise and easy to understand.

Task 3:

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



```python
class BankAccount:
    def __init__(self, account_holder, balance=0):
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposited: {amount}. New balance: {self.balance}")
        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("Insufficient balance. Withdrawal denied.")
        else:
            self.balance -= amount
            print(f"Withdrew: {amount}. New balance: {self.balance}")

# Example usage:
acc = BankAccount("John Doe", 100)
acc.deposit(50)
acc.withdraw(200)
acc.withdraw(30)
```

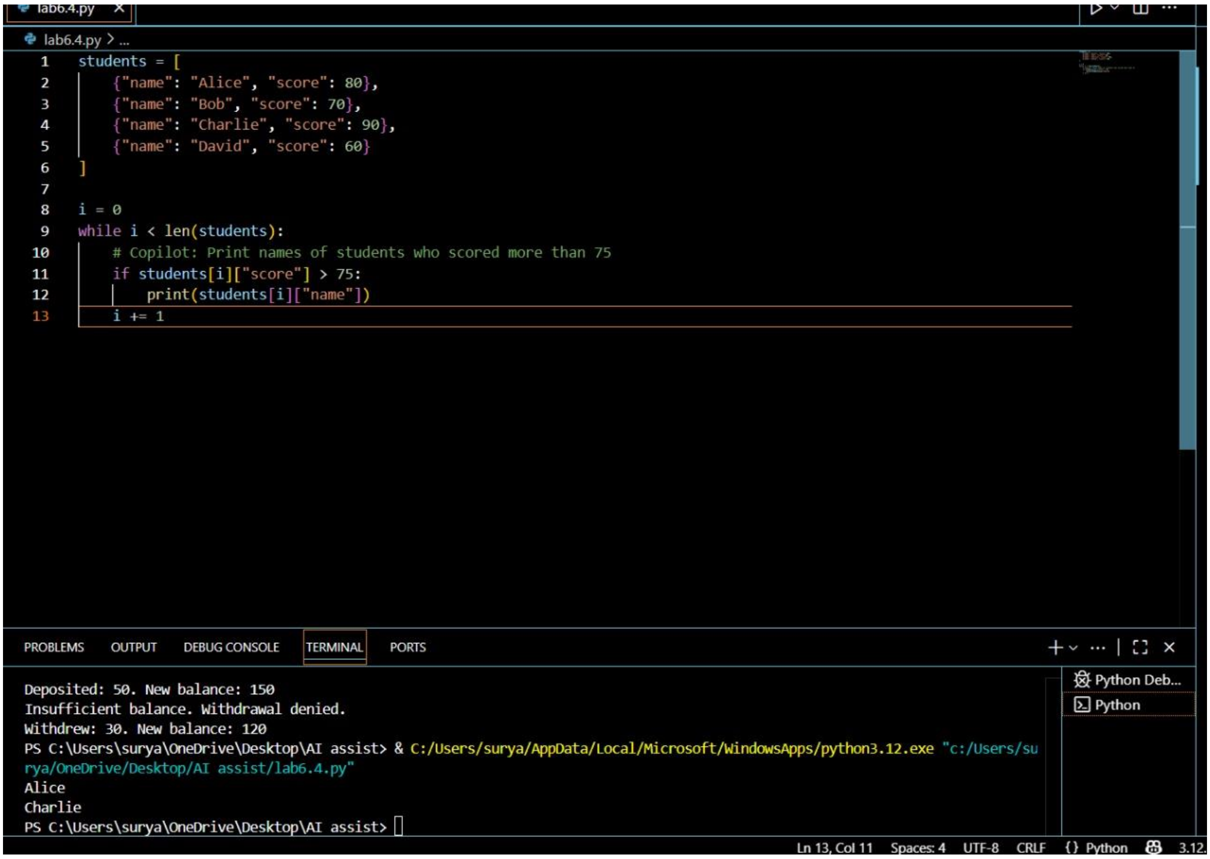
```
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/lab6.4.py"
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/lab6.4.py"
Deposited: 50. New balance: 150
Insufficient balance. Withdrawal denied.
Withdrew: 30. New balance: 120
PS C:\Users\surya\OneDrive\Desktop\AI assist> 
```

Observation:

The BankAccount class lets you deposit and withdraw money, and checks for insufficient balance before withdrawing. The code is simple and easy to understand.

Task 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



```python
students = [
    {"name": "Alice", "score": 80},
    {"name": "Bob", "score": 70},
    {"name": "Charlie", "score": 90},
    {"name": "David", "score": 60}
]

i = 0
while i < len(students):
    # Copilot: Print names of students who scored more than 75
    if students[i]["score"] > 75:
        print(students[i]["name"])
    i += 1
```

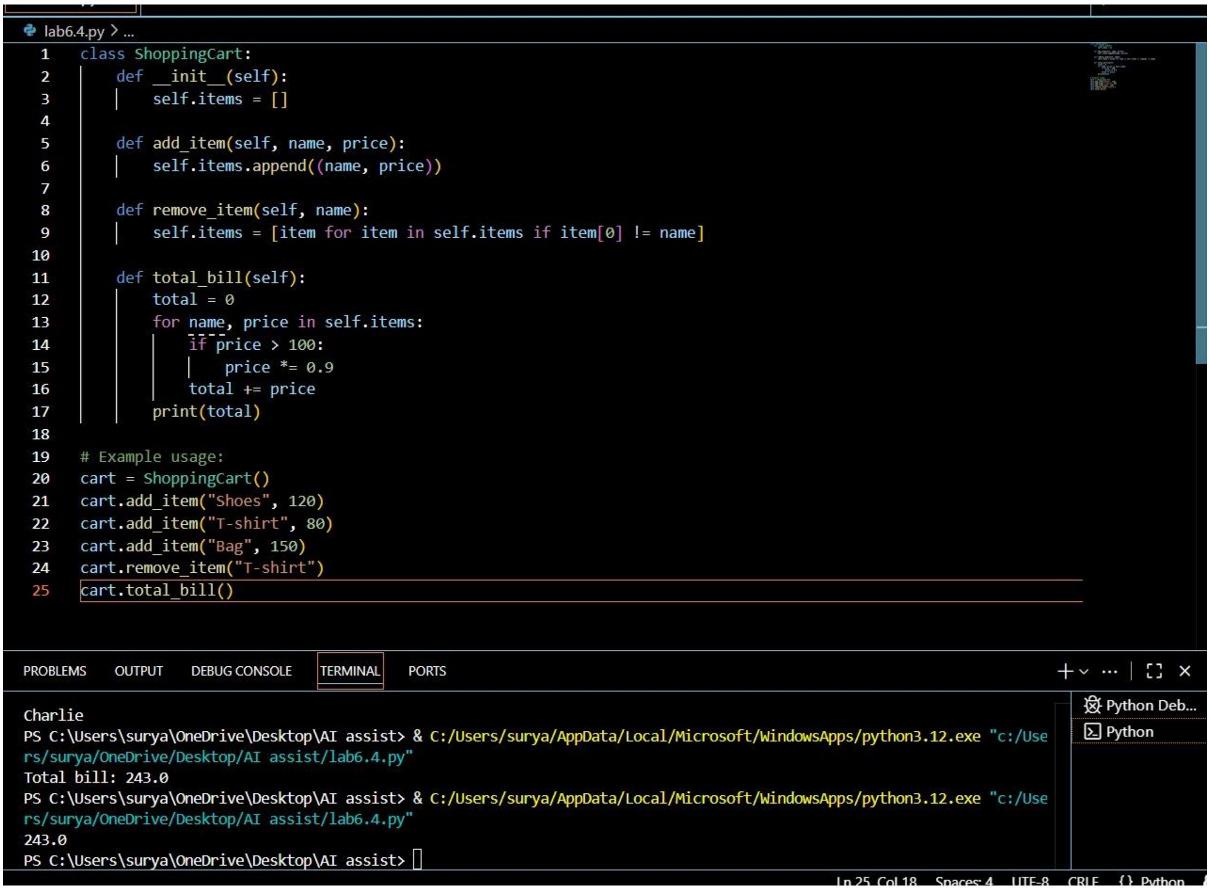
```
Deposited: 50. New balance: 150
Insufficient balance. Withdrawal denied.
Withdrew: 30. New balance: 120
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/su
rya/OneDrive/Desktop/AI assist/lab6.4.py"
Alice
Charlie
PS C:\Users\surya\OneDrive\Desktop\AI assist> 
```

Observation:

A list of student dictionaries stores each student's name and score. The while loop checks each student and prints the names of those who scored more than 75. The code is simple and uses basic list and loop concepts.

Task 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.



```python
class ShoppingCart:
    def __init__(self):
        self.items = []

    def add_item(self, name, price):
        self.items.append((name, price))

    def remove_item(self, name):
        self.items = [item for item in self.items if item[0] != name]

    def total_bill(self):
        total = 0
        for name, price in self.items:
            if price > 100:
                price *= 0.9
            total += price
        print(total)

# Example usage:
cart = ShoppingCart()
cart.add_item("Shoes", 120)
cart.add_item("T-shirt", 80)
cart.add_item("Bag", 150)
cart.remove_item("T-shirt")
cart.total_bill()
```

```
Charlie
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Use
rs/surya/OneDrive/Desktop/AI assist/lab6.4.py"
Total bill: 243.0
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Use
rs/surya/OneDrive/Desktop/AI assist/lab6.4.py"
243.0
PS C:\Users\surya\OneDrive\Desktop\AI assist> 
```

Observation:

The ShoppingCart class starts with an empty items list. It has methods to add and remove items, and uses a loop to calculate the total bill. If an item costs more than 100, a discount is applied. The code is simple and shows basic class and loop usage.