AI Assisted Coding
Assignment - 6.5

Name: L. Sharan Sai Varshith
Ht.no:2303A51450
Bt.no:21

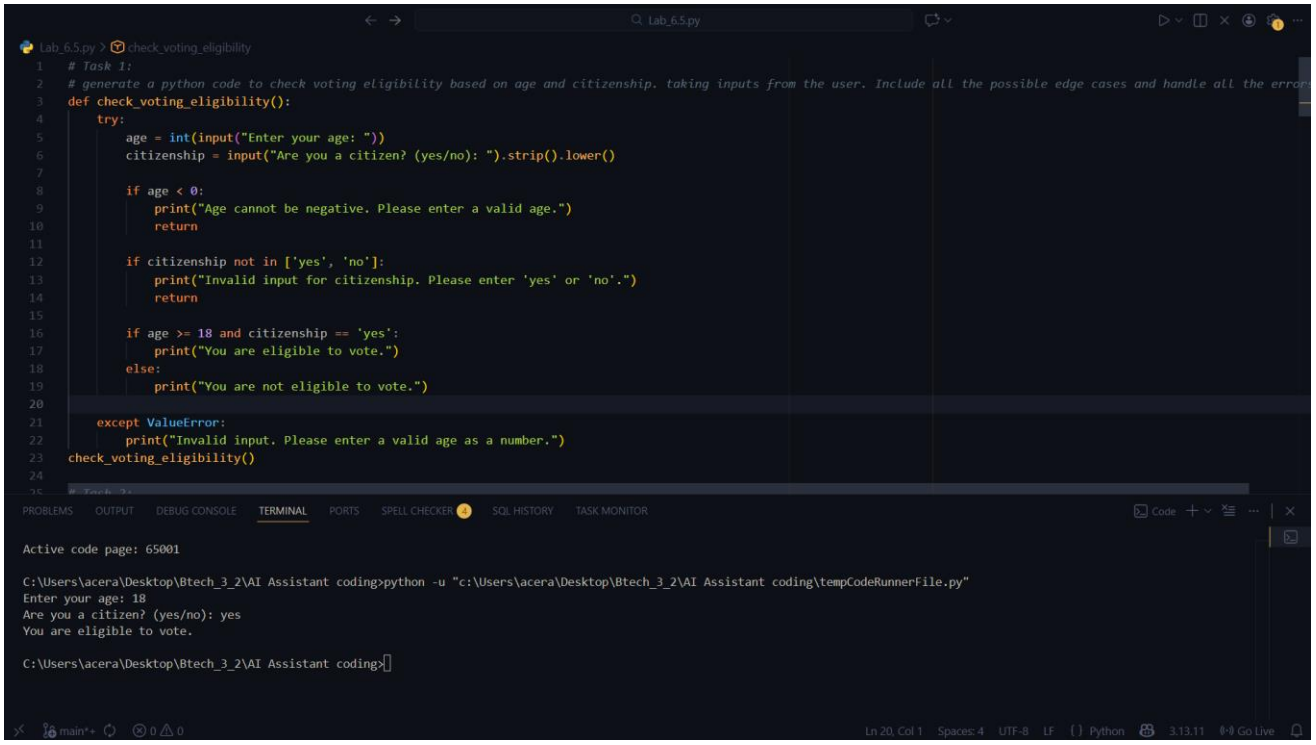
Task Description #1

(AI-Based Code Completion for Conditional Eligibility Check)

Task: Use an AI tool to generate eligibility logic.

Prompt: “Generate Python code to check voting eligibility based on age and citizenship.”

Code:



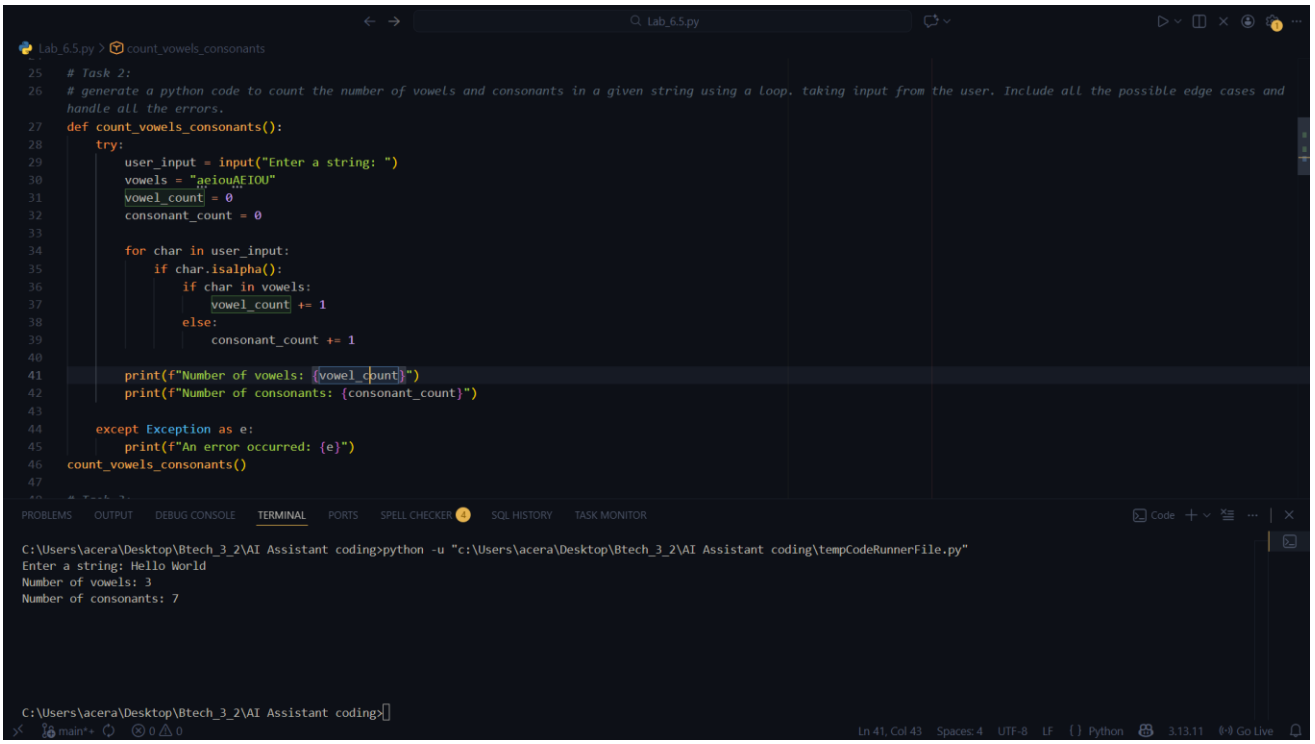
Task Description #2

(AI-Based Code Completion for Loop-Based String Processing)
Task: Use an AI tool to process strings using loops.

Prompt:

"Generate Python code to count vowels and consonants in a string using a loop."

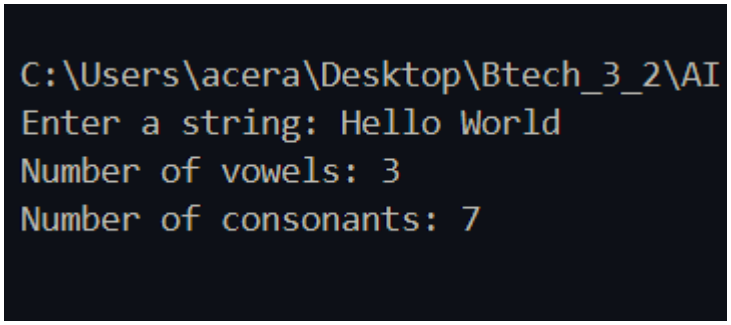
Code:



```
# Task 2:
# generate a python code to count the number of vowels and consonants in a given string using a loop. taking input from the user. Include all the possible edge cases and handle all the errors.
def count_vowels_consonants():
    try:
        user_input = input("Enter a string: ")
        vowels = "aeiouAEIOU"
        vowel_count = 0
        consonant_count = 0

        for char in user_input:
            if char.isalpha():
                if char in vowels:
                    vowel_count += 1
                else:
                    consonant_count += 1

        print(f"Number of vowels: {vowel_count}")
        print(f"Number of consonants: {consonant_count}")

    except Exception as e:
        print(f"An error occurred: {e}")
count_vowels_consonants()
```

```
C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>python -u "c:\Users\acera\Desktop\Btech_3_2\AI Assistant coding\tempCodeRunnerFile.py"
Enter a string: Hello World
Number of vowels: 3
Number of consonants: 7

C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>
```

Output:



```
C:\Users\acera\Desktop\Btech_3_2\AI
Enter a string: Hello World
Number of vowels: 3
Number of consonants: 7
```

Task Description #3

(AI-Assisted Code Completion Reflection Task)

Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

Prompt:

"Generate a Python program for a library management system using classes, loops, and conditional statements."

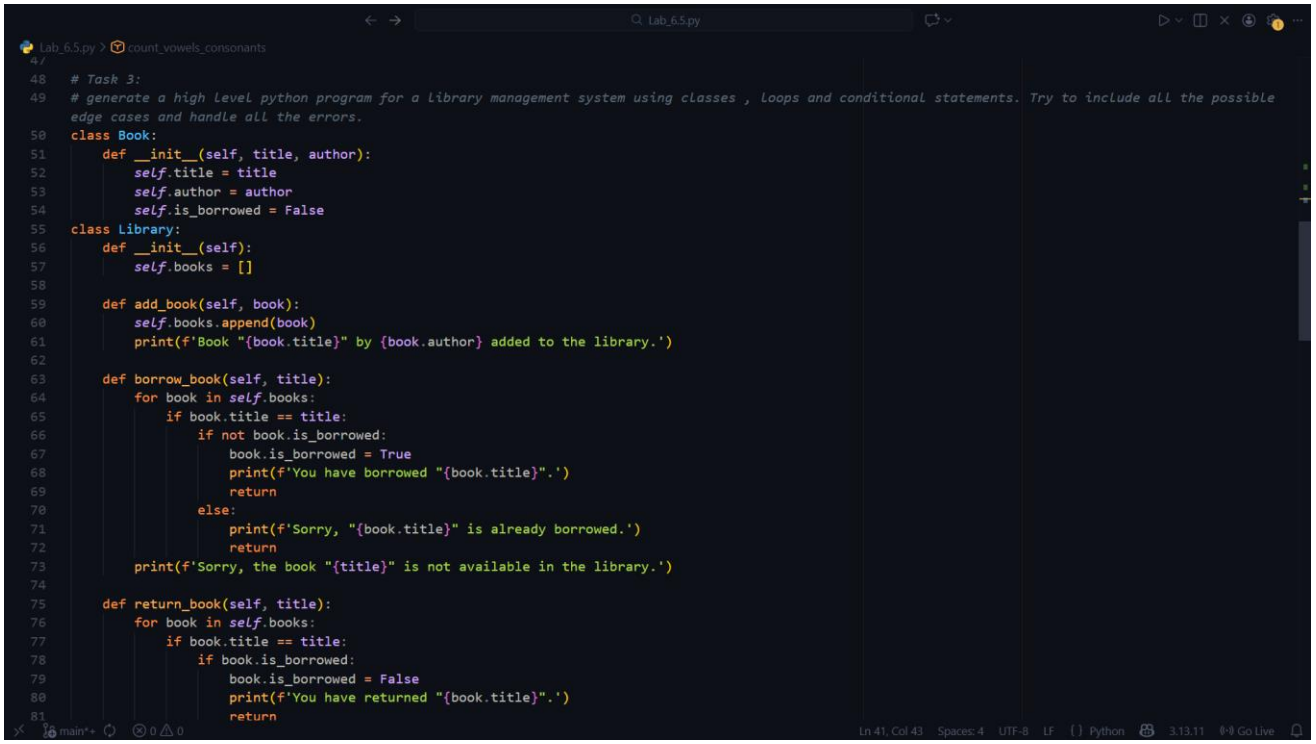
Expected Output:

• Complete AI-generated program.

• Review of AI suggestions quality.

• Short reflection on AI-assisted coding experience.

Code:



```python
# Task 3:
# generate a high level python program for a library management system using classes , loops and conditional statements. Try to include all the possible edge cases and handle all the errors.
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_borrowed = False
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)
        print(f'Book "{book.title}" by {book.author} added to the library.')

    def borrow_book(self, title):
        for book in self.books:
            if book.title == title:
                if not book.is_borrowed:
                    book.is_borrowed = True
                    print(f'You have borrowed "{book.title}".')
                    return
                else:
                    print(f'Sorry, "{book.title}" is already borrowed.')
                    return
        print(f'Sorry, the book "{title}" is not available in the library.')

    def return_book(self, title):
        for book in self.books:
            if book.title == title:
                if book.is_borrowed:
                    book.is_borrowed = False
                    print(f'You have returned "{book.title}".')
                    return
```

Output:

```
C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>python -u "c:\Users\acera\Desktop\Btech_3_2\AI Assistant coding\tempCodeRunnerFile.py"

Library Management System
1. Add Book
2. Borrow Book
3. Return Book
4. Exit
Enter your choice (1-4): 1
Enter book title: Tales
Enter book author: Herculus
Book "Tales" by Herculus added to the library.

Library Management System
1. Add Book
2. Borrow Book
3. Return Book
4. Exit
Enter your choice (1-4): 4
Exiting the Library Management System.
```

Task Description #4

(AI-Assisted Code Completion for Class-Based Attendance System)

Task: Use an AI tool to generate an attendance management class.

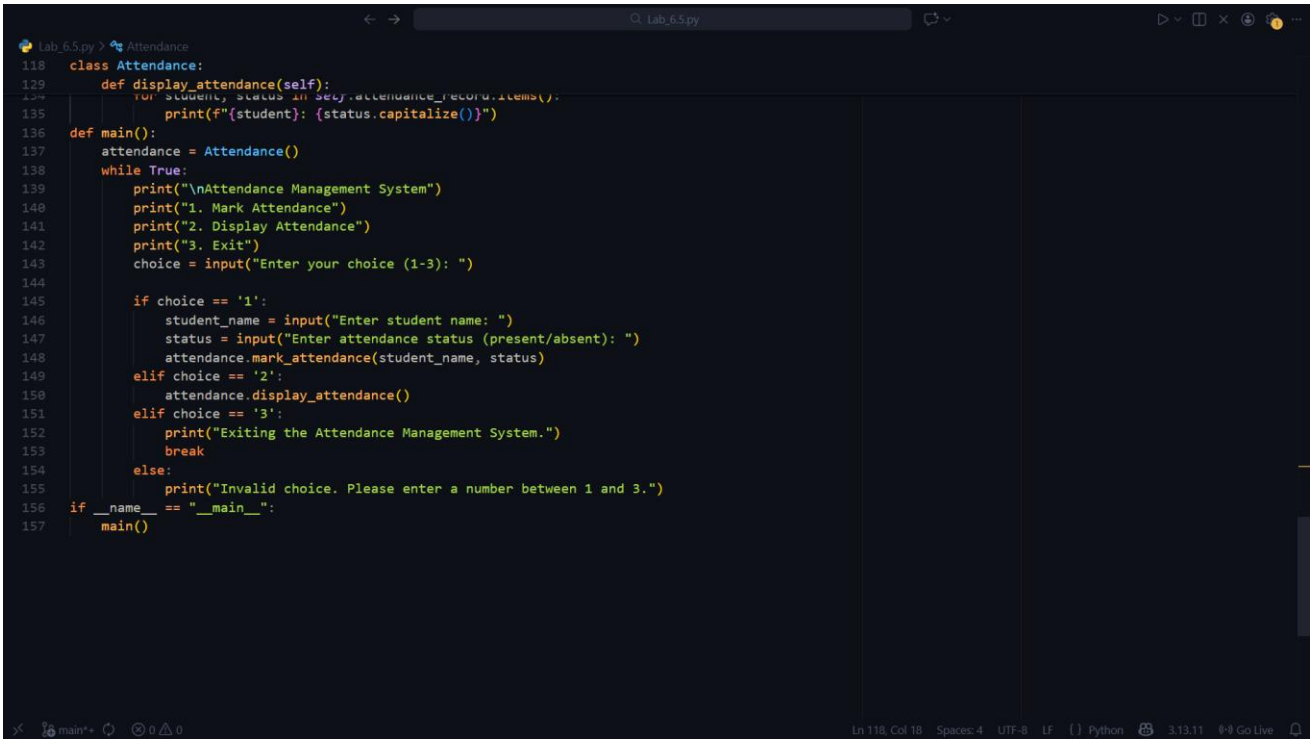
Prompt: “Generate a Python class to mark and display student attendance using loops.”

Expected Output:

- AI-generated attendance logic.
- Correct display of attendance.
- Test cases.

Code:

```python
# Task 4:
# generate a python class to mark and display attendance of the students using loops and conditional statements. Include all the possible edge cases and handle all the errors.
class Attendance:
    def __init__(self):
        self.attendance_record = {}

    def mark_attendance(self, student_name, status):
        if status.lower() not in ['present', 'absent']:
            print("Invalid status. Please enter 'present' or 'absent'.")
            return
        self.attendance_record[student_name] = status.lower()
        print(f"Attendance marked for {student_name} as {status}.")

    def display_attendance(self):
        if not self.attendance_record:
            print("No attendance records available.")
            return
        print("\nAttendance Record:")
        for student, status in self.attendance_record.items():
            print(f"{student}: {status.capitalize()}")
def main():
    attendance = Attendance()
    while True:
        print("\nAttendance Management System")
        print("1. Mark Attendance")
        print("2. Display Attendance")
        print("3. Exit")
        choice = input("Enter your choice (1-3): ")

        if choice == '1':
            student_name = input("Enter student name: ")
            status = input("Enter attendance status (present/absent): ")
            attendance.mark_attendance(student_name, status)
        elif choice == '2':
```



```python
class Attendance:
    def display_attendance(self):
            print(f"{student}: {status.capitalize()}")
def main():
    attendance = Attendance()
    while True:
        print("\nAttendance Management System")
        print("1. Mark Attendance")
        print("2. Display Attendance")
        print("3. Exit")
        choice = input("Enter your choice (1-3): ")

        if choice == '1':
            student_name = input("Enter student name: ")
            status = input("Enter attendance status (present/absent): ")
            attendance.mark_attendance(student_name, status)
        elif choice == '2':
            attendance.display_attendance()
        elif choice == '3':
            print("Exiting the Attendance Management System.")
            break
        else:
            print("Invalid choice. Please enter a number between 1 and 3.")
if __name__ == "__main__":
    main()
```

Output:

```
Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 1
Enter student name: sharan
Enter attendance status (present/absent): present
Attendance marked for sharan as present.

Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 2

Attendance Record:
sharan: Present

Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 3
Exiting the Attendance Management System.
```

Task Description #5

(AI-Based Code Completion for Conditional Menu Navigation)

Task: Use an AI tool to complete a navigation menu.

Prompt: “Generate a Python program using loops and conditionals to simulate an ATM menu.”

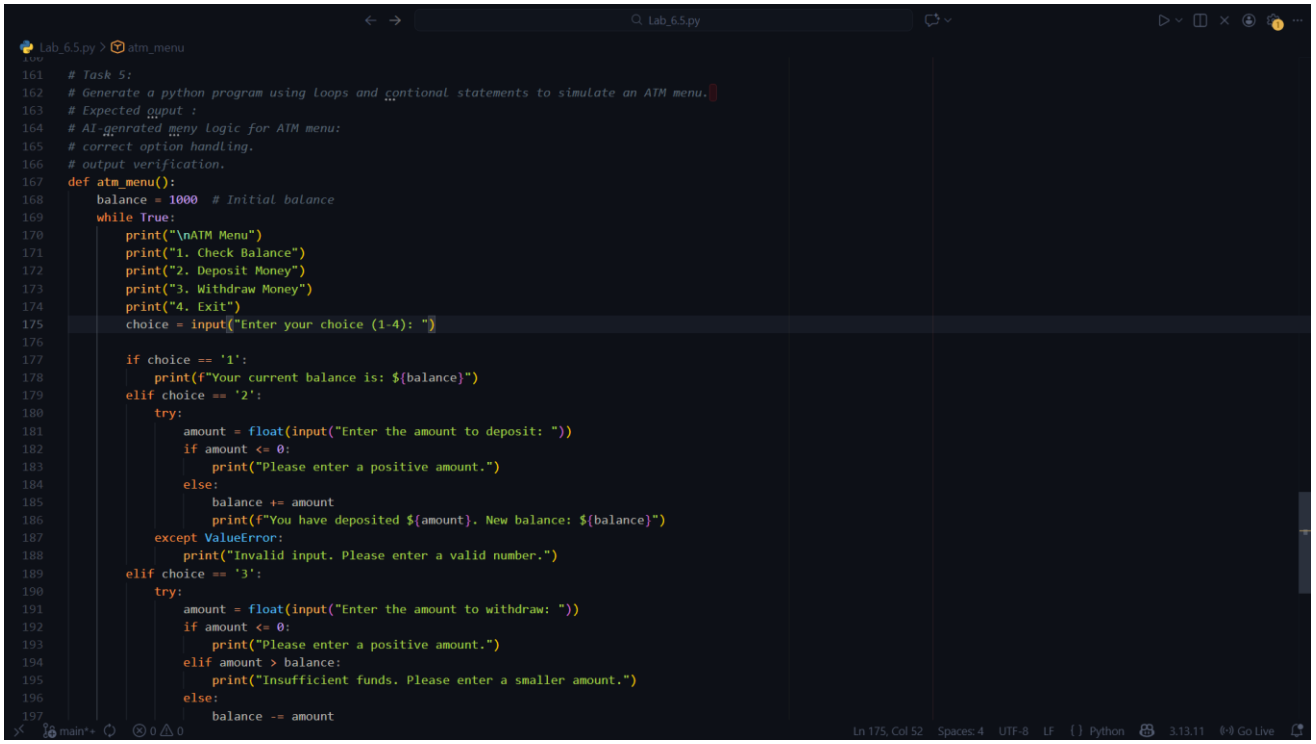
Expected Output:

• AI-generated menu logic.

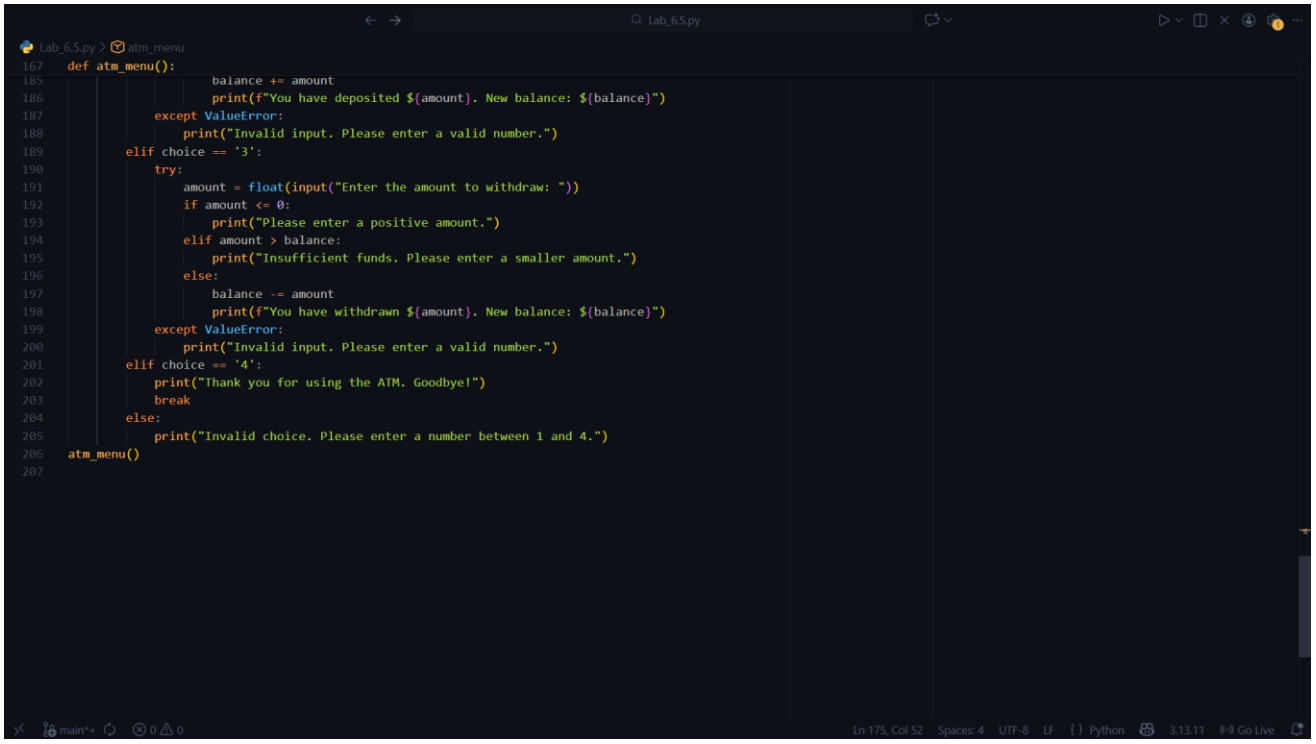
• Correct option handling.

• Output verification

Code:

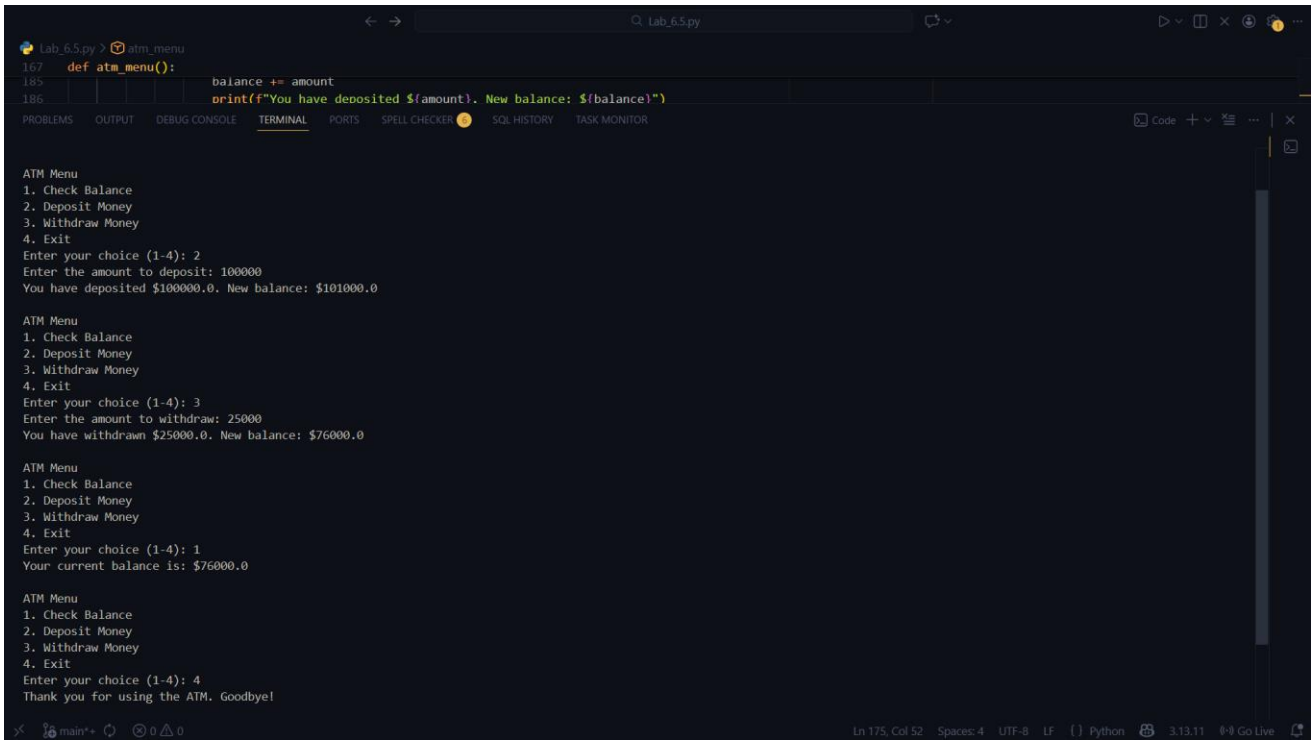


```python
# Task 5:
# Generate a python program using loops and contional statements to simulate an ATM menu.
# Expected ouput :
# AI-genrated meny logic for ATM menu:
# correct option handling.
# output verification.
def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print("\nATM Menu")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Exit")
        choice = input("Enter your choice (1-4): ")

        if choice == '1':
            print(f"Your current balance is: ${balance}")
        elif choice == '2':
            try:
                amount = float(input("Enter the amount to deposit: "))
                if amount <= 0:
                    print("Please enter a positive amount.")
                else:
                    balance += amount
                    print(f"You have deposited ${amount}. New balance: ${balance}")
            except ValueError:
                print("Invalid input. Please enter a valid number.")
        elif choice == '3':
            try:
                amount = float(input("Enter the amount to withdraw: "))
                if amount <= 0:
                    print("Please enter a positive amount.")
                elif amount > balance:
                    print("Insufficient funds. Please enter a smaller amount.")
                else:
                    balance -= amount
```



```python
def atm_menu():
                    balance += amount
                    print(f"You have deposited ${amount}. New balance: ${balance}")
            except ValueError:
                print("Invalid input. Please enter a valid number.")
        elif choice == '3':
            try:
                amount = float(input("Enter the amount to withdraw: "))
                if amount <= 0:
                    print("Please enter a positive amount.")
                elif amount > balance:
                    print("Insufficient funds. Please enter a smaller amount.")
                else:
                    balance -= amount
                    print(f"You have withdrawn ${amount}. New balance: ${balance}")
            except ValueError:
                print("Invalid input. Please enter a valid number.")
        elif choice == '4':
            print("Thank you for using the ATM. Goodbye!")
            break
        else:
            print("Invalid choice. Please enter a number between 1 and 4.")
atm_menu()
```

Output:


Lab_6.5.py
Lab_6.5.py > atm_menu
167 def atm_menu():
185 balance += amount
186 print(f"You have deposited ${amount}. New balance: ${balance}")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER SQL HISTORY TASK MONITOR
Code
ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 2
Enter the amount to deposit: 100000
You have deposited $100000.0. New balance: $101000.0
ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 3
Enter the amount to withdraw: 25000
You have withdrawn $25000.0. New balance: $76000.0
ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 1
Your current balance is: $76000.0
ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 4
Thank you for using the ATM. Goodbye!
main*+ 0 0
Ln 175, Col 52 Spaces: 4 UTF-8 LF Python 3.13.11 Go Live