AI ASSISTANT CODING

Assignment – 6.5

Name: K. Shailaja

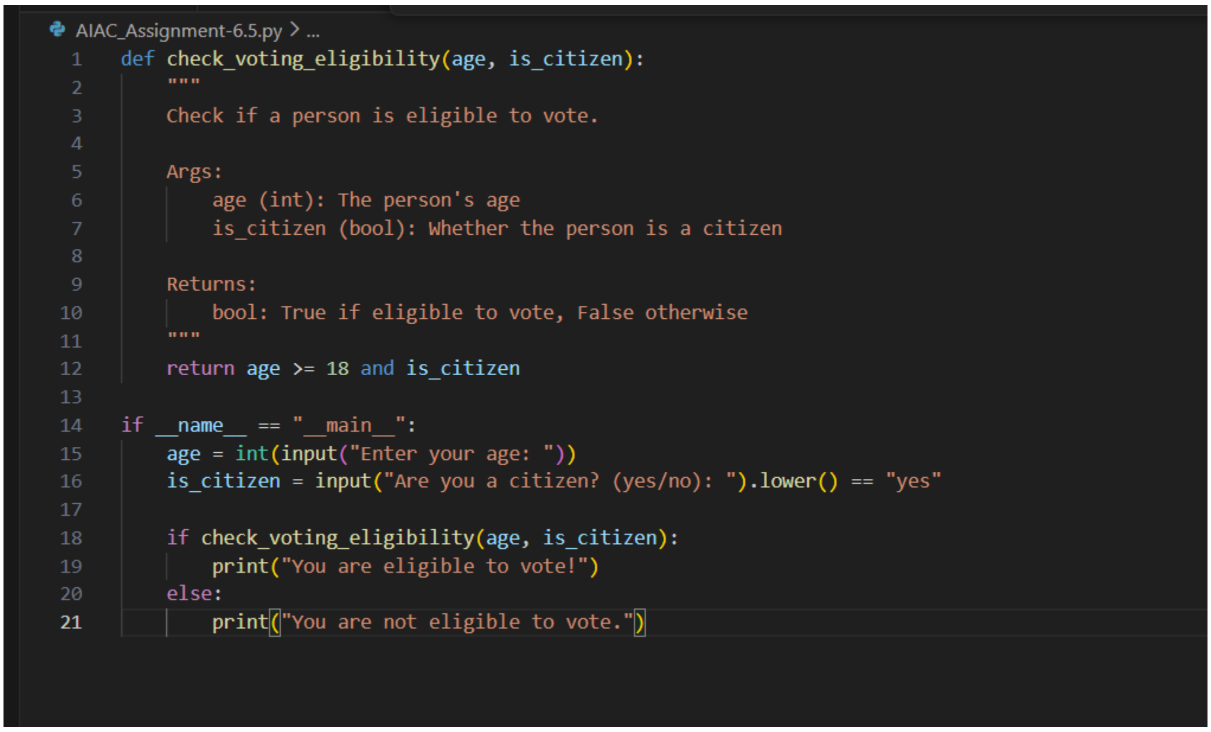
H.T No.: 2303A52155

Batch: 34

Question:

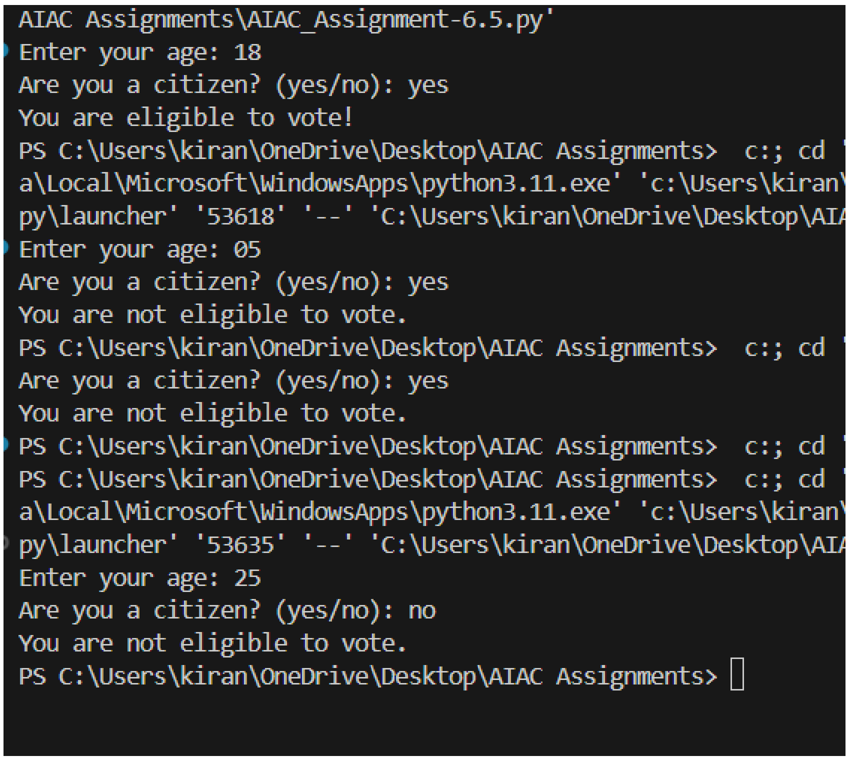
Experiment 6: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals Week3 - Friday

LO1. Use AI-based code completion tools to generate Python code involving classes, loops, and conditionals.
LO2. Interpret and explain AI-generated code line-by-line.
LO3. Identify errors, inefficiencies, or logical flaws in AI-suggested implementations.
LO4. Optimize AI-generated code for better readability and performance.
LO5. Demonstrate ethical and responsible use of AI tools in coding tasks.
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.
Expected Output:
• AI-generated conditional logic.



```
AIAC_Assignment-6.5.py > ...
def check_voting_eligibility(age, is_citizen):
    """
    Check if a person is eligible to vote.

    Args:
        age (int): The person's age
        is_citizen (bool): Whether the person is a citizen

    Returns:
        bool: True if eligible to vote, False otherwise
    """
    return age >= 18 and is_citizen

if __name__ == "__main__":
    age = int(input("Enter your age: "))
    is_citizen = input("Are you a citizen? (yes/no): ").lower() == "yes"

    if check_voting_eligibility(age, is_citizen):
        print("You are eligible to vote!")
    else:
        print("You are not eligible to vote.")
```

- **Correct eligibility decisions.**

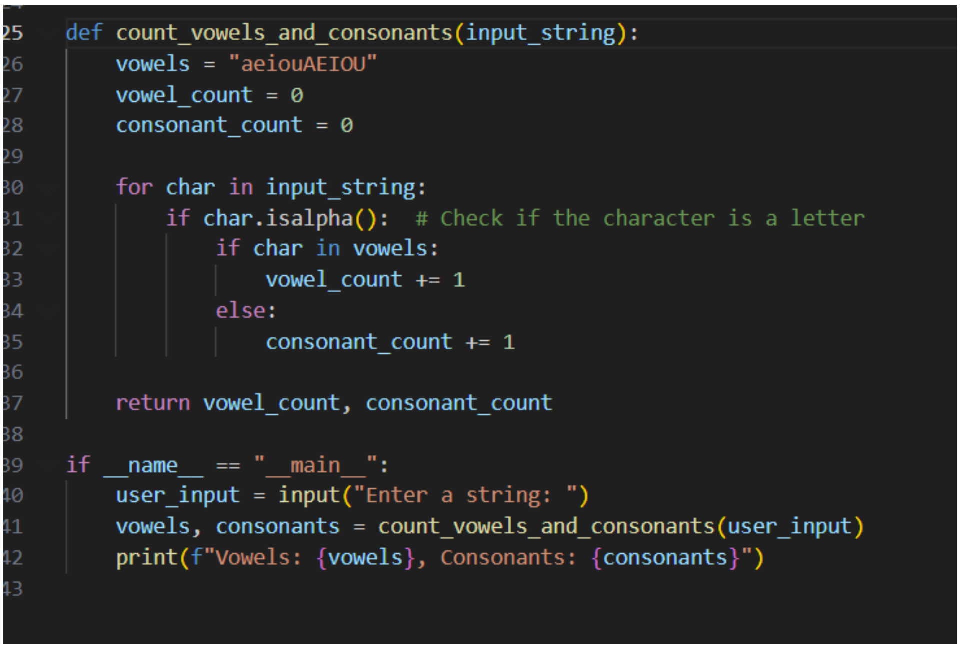


```
AIAC Assignments\AIAC_Assignment-6.5.py'
Enter your age: 18
Are you a citizen? (yes/no): yes
You are eligible to vote!
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>  c:; cd
a\Local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\kiran
py\launcher' '53618' '--' 'C:\Users\kiran\OneDrive\Desktop\AI
Enter your age: 05
Are you a citizen? (yes/no): yes
You are not eligible to vote.
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>  c:; cd
Are you a citizen? (yes/no): yes
You are not eligible to vote.
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>  c:; cd
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>  c:; cd
a\Local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\kiran
py\launcher' '53635' '--' 'C:\Users\kiran\OneDrive\Desktop\AI
Enter your age: 25
Are you a citizen? (yes/no): no
You are not eligible to vote.
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>
```

- **Explanation of conditions.**

Voting eligibility is checked using Python conditional statements based on two conditions. First, the person must be 18 years or older, verified using age >= 18. Second, the person must be a citizen, checked using a Boolean value (True or False). Both conditions are combined using the logical AND (and) operator. If both conditions are satisfied, the output is “Eligible to vote”; otherwise, the program displays “Not eligible to vote.”

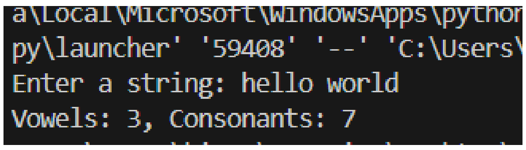
Task Description 2(AI-Based Code Completion for Loop-Based String Processing)
Task: Use an AI tool to process strings using loops.
Prompt:
Generate Python code to count vowels and consonants in a string using a loop.
Expected Output:
• AI-generated string processing logic.



```python
def count_vowels_and_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if char.isalpha():  # Check if the character is a letter
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count

if __name__ == "__main__":
    user_input = input("Enter a string: ")
    vowels, consonants = count_vowels_and_consonants(user_input)
    print(f"Vowels: {vowels}, Consonants: {consonants}")
```

• Correct counts.

The program accurately counts vowels and consonants by looping through each character, considering only alphabetic characters, counting vowels separately, and treating all other alphabets as consonants.
• Output verification.



```
a\Local\Microsoft\WindowsApps\python
py\launcher' '59408' '--' 'C:\Users\
Enter a string: hello world
Vowels: 3, Consonants: 7
```

Task Description 3 (AI-Assisted Code Completion Reflection Task)

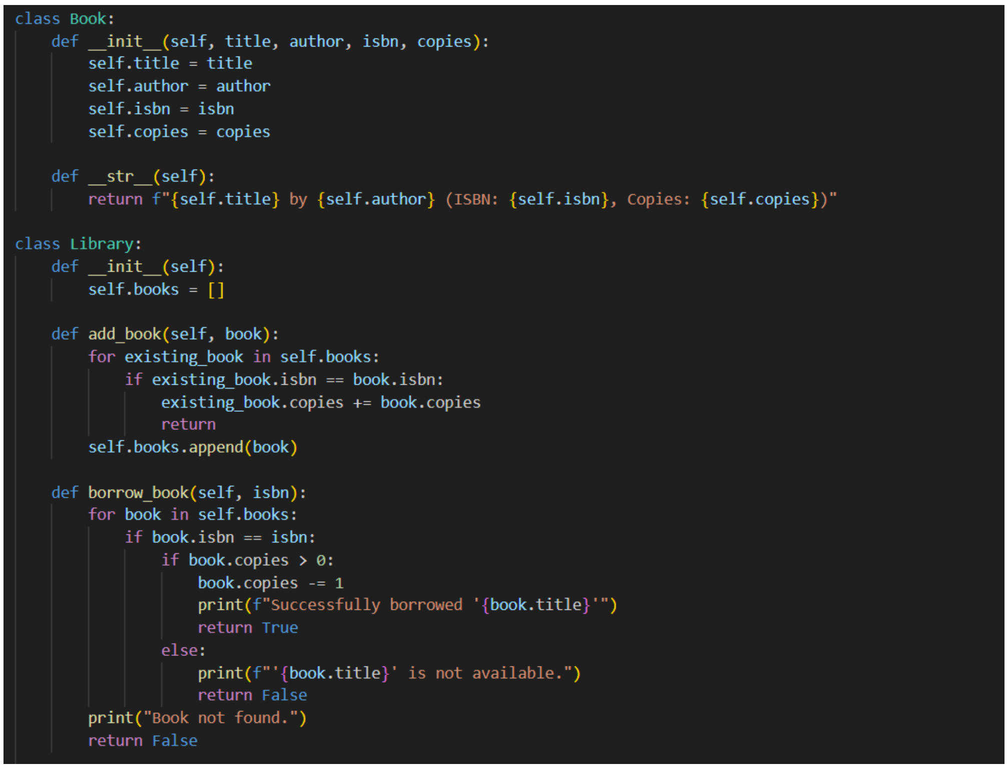
Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

Prompt:

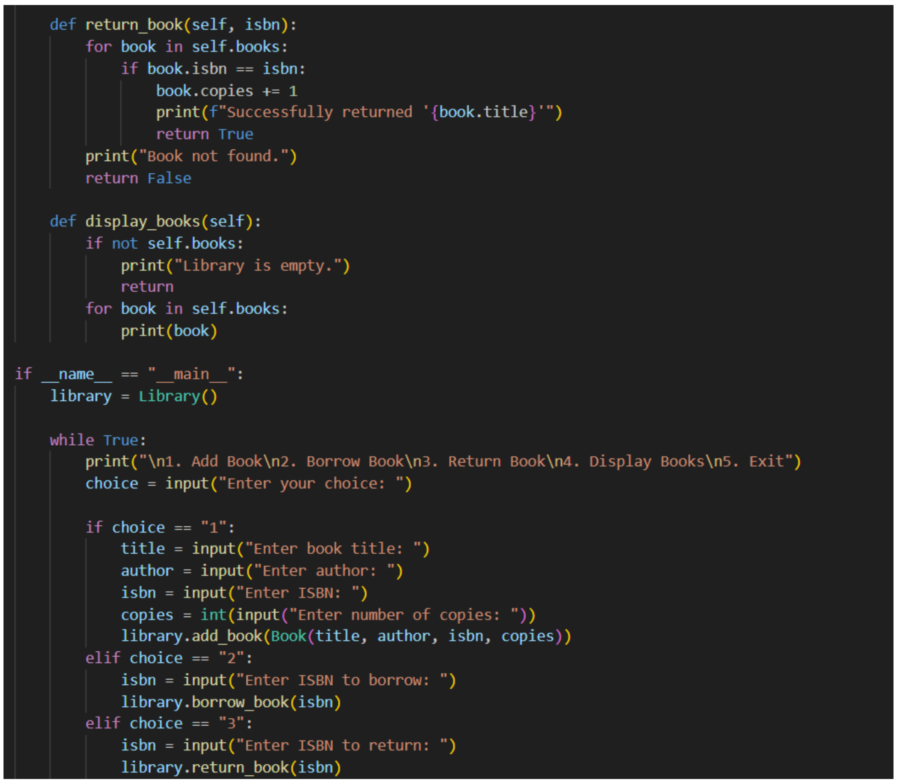
Generate a Python program for a library management system using classes, loops, and conditional statements.

Expected Output:

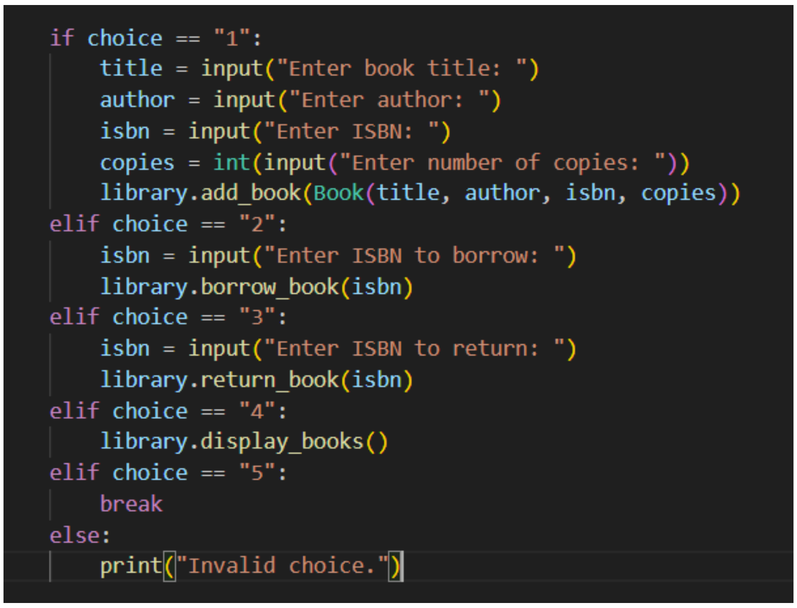
• Complete AI-generated program.



```python
class Book:
    def __init__(self, title, author, isbn, copies):
        self.title = title
        self.author = author
        self.isbn = isbn
        self.copies = copies

    def __str__(self):
        return f"{self.title} by {self.author} (ISBN: {self.isbn}, Copies: {self.copies})"

class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        for existing_book in self.books:
            if existing_book.isbn == book.isbn:
                existing_book.copies += book.copies
                return
        self.books.append(book)

    def borrow_book(self, isbn):
        for book in self.books:
            if book.isbn == isbn:
                if book.copies > 0:
                    book.copies -= 1
                    print(f"Successfully borrowed '{book.title}'")
                    return True
                else:
                    print(f"'{book.title}' is not available.")
                    return False
        print("Book not found.")
        return False
```



```python
    def return_book(self, isbn):
        for book in self.books:
            if book.isbn == isbn:
                book.copies += 1
                print(f"Successfully returned '{book.title}'")
                return True
        print("Book not found.")
        return False

    def display_books(self):
        if not self.books:
            print("Library is empty.")
            return
        for book in self.books:
            print(book)

if __name__ == "__main__":
    library = Library()

    while True:
        print("\n1. Add Book\n2. Borrow Book\n3. Return Book\n4. Display Books\n5. Exit")
        choice = input("Enter your choice: ")

        if choice == "1":
            title = input("Enter book title: ")
            author = input("Enter author: ")
            isbn = input("Enter ISBN: ")
            copies = int(input("Enter number of copies: "))
            library.add_book(Book(title, author, isbn, copies))
        elif choice == "2":
            isbn = input("Enter ISBN to borrow: ")
            library.borrow_book(isbn)
        elif choice == "3":
            isbn = input("Enter ISBN to return: ")
            library.return_book(isbn)
```



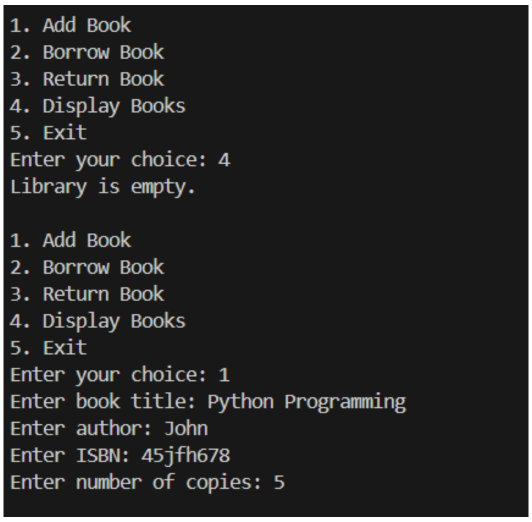
```python
if choice == "1":
    title = input("Enter book title: ")
    author = input("Enter author: ")
    isbn = input("Enter ISBN: ")
    copies = int(input("Enter number of copies: "))
    library.add_book(Book(title, author, isbn, copies))
elif choice == "2":
    isbn = input("Enter ISBN to borrow: ")
    library.borrow_book(isbn)
elif choice == "3":
    isbn = input("Enter ISBN to return: ")
    library.return_book(isbn)
elif choice == "4":
    library.display_books()
elif choice == "5":
    break
else:
    print("Invalid choice.")
```

• **Review of AI suggestions quality.**

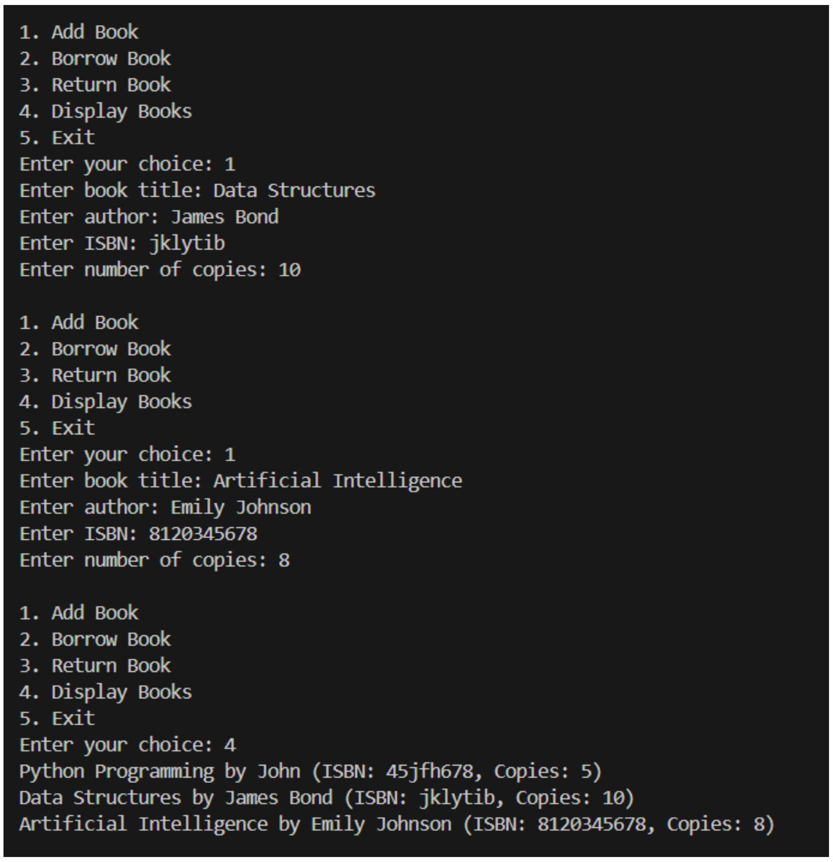
The AI-generated code is well-structured, easy to understand, and correctly uses object-oriented concepts, loops, and conditionals. It’s clear and functional logic makes it suitable for beginners and academic use.

• **Short reflection on AI-assisted coding experience.**

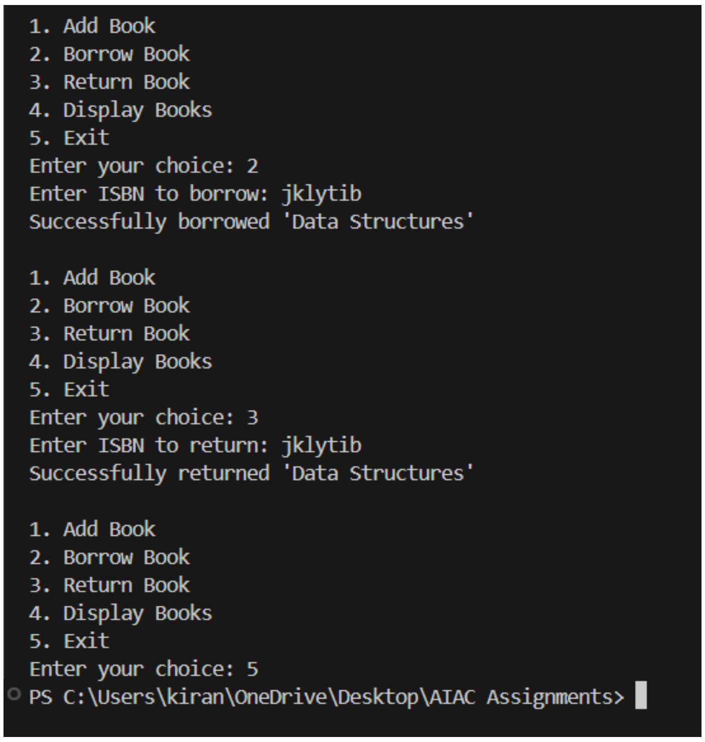
AI-assisted coding helped quickly generate a complete and working program, saving time and providing a clear structure. It supports learning programming concepts and improves problem-solving efficiency.



```
1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 4
Library is empty.

1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 1
Enter book title: Python Programming
Enter author: John
Enter ISBN: 45jfh678
Enter number of copies: 5
```



```
1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 1
Enter book title: Data Structures
Enter author: James Bond
Enter ISBN: jklytib
Enter number of copies: 10

1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 1
Enter book title: Artificial Intelligence
Enter author: Emily Johnson
Enter ISBN: 8120345678
Enter number of copies: 8

1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 4
Python Programming by John (ISBN: 45jfh678, Copies: 5)
Data Structures by James Bond (ISBN: jklytib, Copies: 10)
Artificial Intelligence by Emily Johnson (ISBN: 8120345678, Copies: 8)
```



```
1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 2
Enter ISBN to borrow: jklytib
Successfully borrowed 'Data Structures'

1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 3
Enter ISBN to return: jklytib
Successfully returned 'Data Structures'

1. Add Book
2. Borrow Book
3. Return Book
4. Display Books
5. Exit
Enter your choice: 5
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>
```

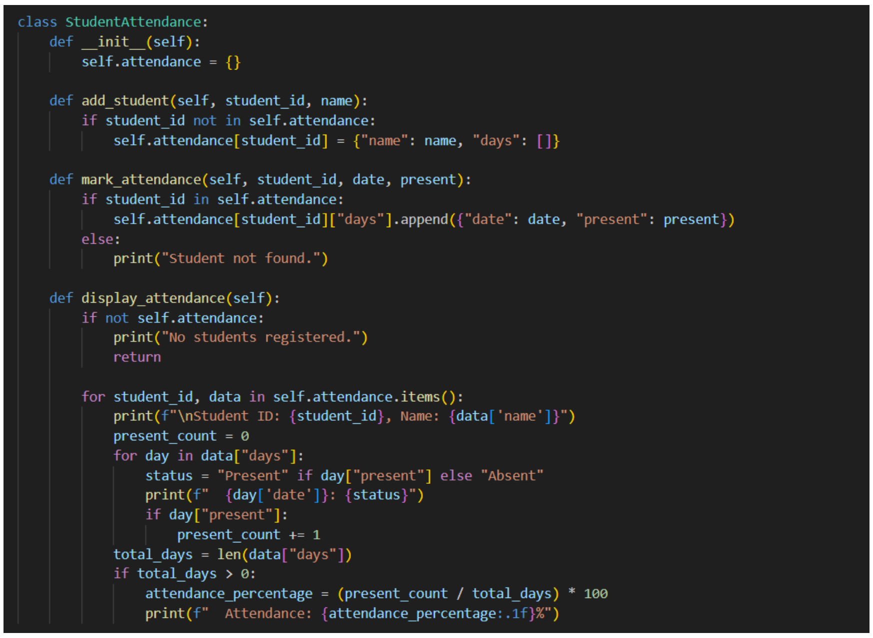
Task Description 4 (AI-Assisted Code Completion for Class-Based Attendance System)

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

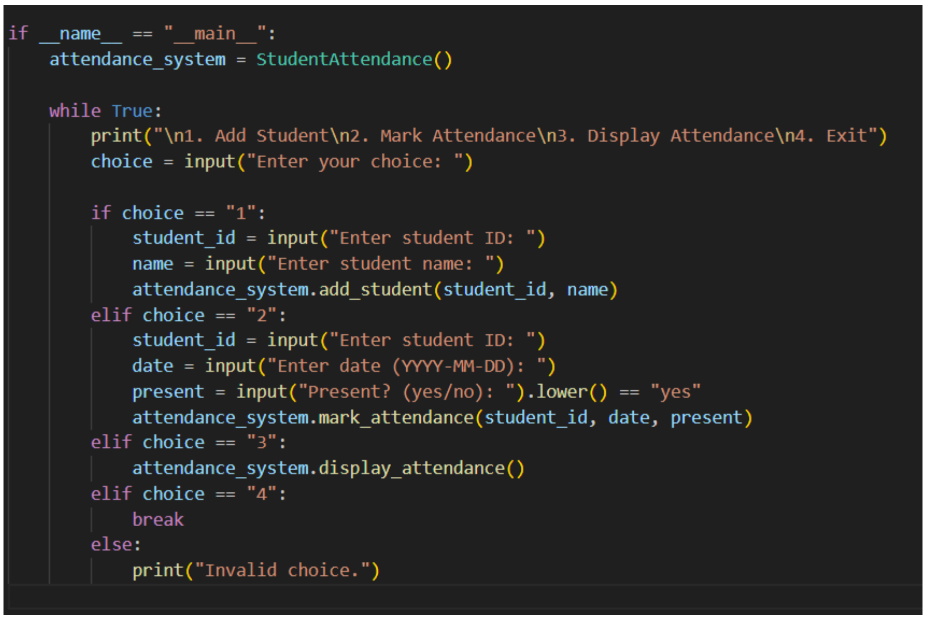
Prompt: Generate a Python class to mark and display student attendance using loops.

Expected Output:

- **AI-generated attendance logic.**



```python
class StudentAttendance:
    def __init__(self):
        self.attendance = {}

    def add_student(self, student_id, name):
        if student_id not in self.attendance:
            self.attendance[student_id] = {"name": name, "days": []}

    def mark_attendance(self, student_id, date, present):
        if student_id in self.attendance:
            self.attendance[student_id]["days"].append({"date": date, "present": present})
        else:
            print("Student not found.")

    def display_attendance(self):
        if not self.attendance:
            print("No students registered.")
            return

        for student_id, data in self.attendance.items():
            print(f"\nStudent ID: {student_id}, Name: {data['name']}")
            present_count = 0
            for day in data["days"]:
                status = "Present" if day["present"] else "Absent"
                print(f"  {day['date']}: {status}")
                if day["present"]:
                    present_count += 1
            total_days = len(data["days"])
            if total_days > 0:
                attendance_percentage = (present_count / total_days) * 100
                print(f"  Attendance: {attendance_percentage:.1f}%")
```

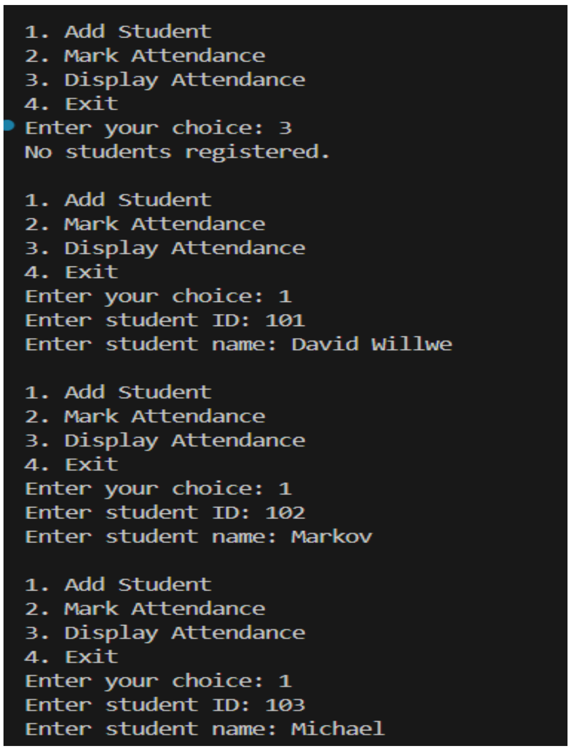


```python
if __name__ == "__main__":
    attendance_system = StudentAttendance()

    while True:
        print("\n1. Add Student\n2. Mark Attendance\n3. Display Attendance\n4. Exit")
        choice = input("Enter your choice: ")

        if choice == "1":
            student_id = input("Enter student ID: ")
            name = input("Enter student name: ")
            attendance_system.add_student(student_id, name)
        elif choice == "2":
            student_id = input("Enter student ID: ")
            date = input("Enter date (YYYY-MM-DD): ")
            present = input("Present? (yes/no): ").lower() == "yes"
            attendance_system.mark_attendance(student_id, date, present)
        elif choice == "3":
            attendance_system.display_attendance()
        elif choice == "4":
            break
        else:
            print("Invalid choice.")
```

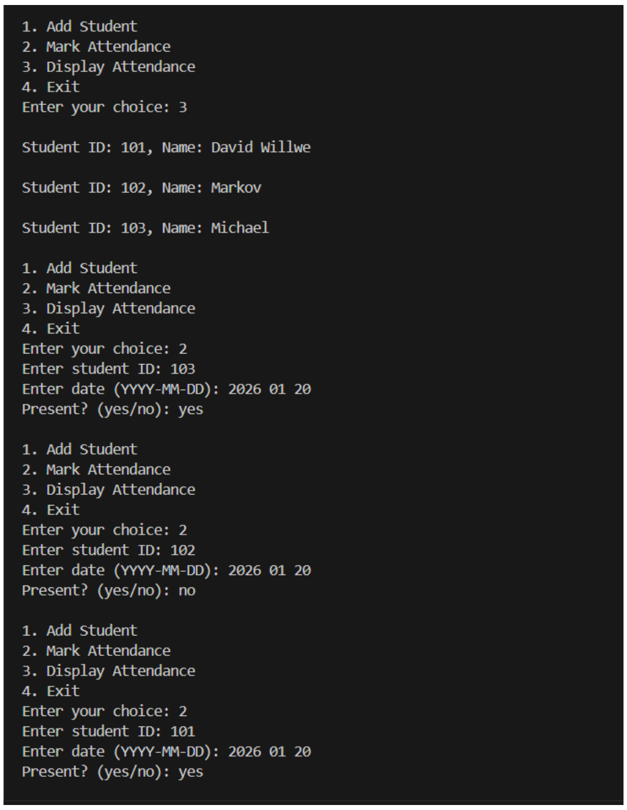
- **Correct display of attendance.**

The attendance records are displayed clearly by listing each student along with their attendance status. If no records are present, the program informs the user accordingly.

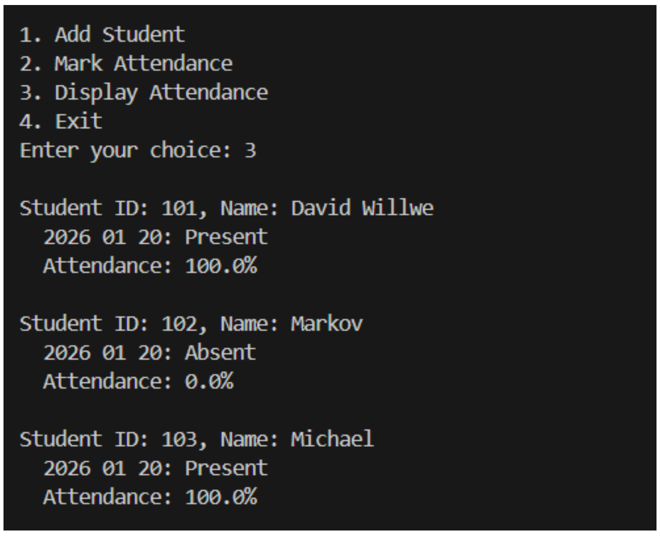
- **Test cases.**



```
1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 3
No students registered.

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 1
Enter student ID: 101
Enter student name: David Willwe

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 1
Enter student ID: 102
Enter student name: Markov

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 1
Enter student ID: 103
Enter student name: Michael
```



```
1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 3

Student ID: 101, Name: David Willwe

Student ID: 102, Name: Markov

Student ID: 103, Name: Michael

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 2
Enter student ID: 103
Enter date (YYYY-MM-DD): 2026 01 20
Present? (yes/no): yes

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 2
Enter student ID: 102
Enter date (YYYY-MM-DD): 2026 01 20
Present? (yes/no): no

1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 2
Enter student ID: 101
Enter date (YYYY-MM-DD): 2026 01 20
Present? (yes/no): yes
```



```
1. Add Student
2. Mark Attendance
3. Display Attendance
4. Exit
Enter your choice: 3

Student ID: 101, Name: David Willwe
  2026 01 20: Present
  Attendance: 100.0%

Student ID: 102, Name: Markov
  2026 01 20: Absent
  Attendance: 0.0%

Student ID: 103, Name: Michael
  2026 01 20: Present
  Attendance: 100.0%
```

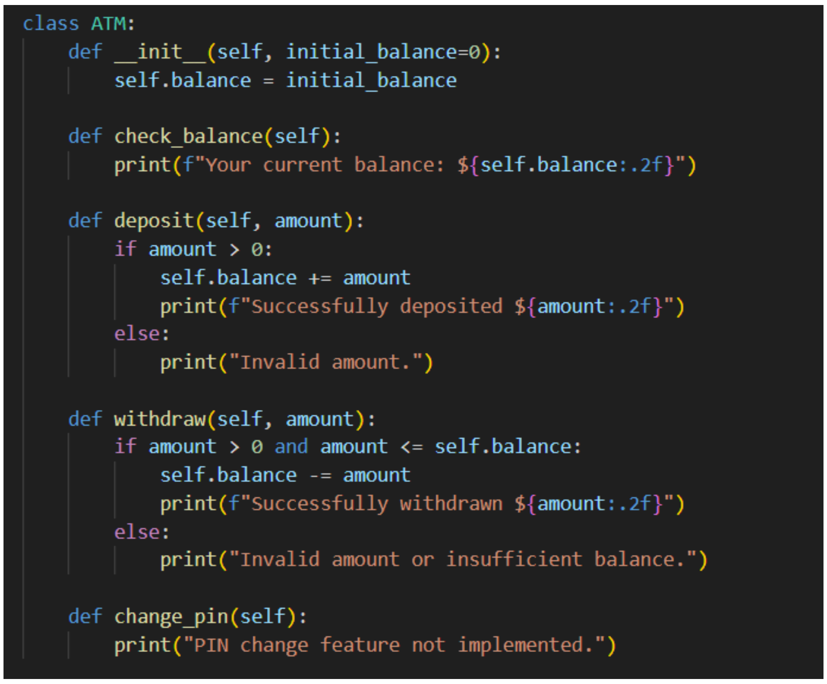
Task Description 5 (AI-Based Code Completion for Conditional Menu Navigation)

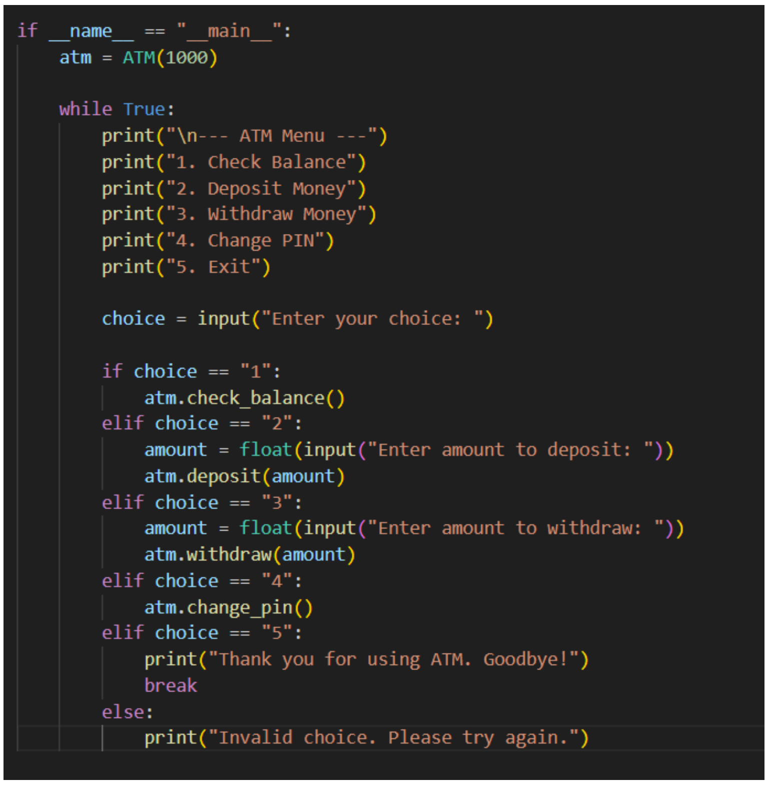
Task: Use an AI tool to complete a navigation menu.

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

Expected Output:

- **AI-generated menu logic.**


```python
class ATM:
    def __init__(self, initial_balance=0):
        self.balance = initial_balance

    def check_balance(self):
        print(f"Your current balance: ${self.balance:.2f}")

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Successfully deposited ${amount:.2f}")
        else:
            print("Invalid amount.")

    def withdraw(self, amount):
        if amount > 0 and amount <= self.balance:
            self.balance -= amount
            print(f"Successfully withdrawn ${amount:.2f}")
        else:
            print("Invalid amount or insufficient balance.")

    def change_pin(self):
        print("PIN change feature not implemented.")
```


```python
if __name__ == "__main__":
    atm = ATM(1000)

    while True:
        print("\n--- ATM Menu ---")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Change PIN")
        print("5. Exit")

        choice = input("Enter your choice: ")

        if choice == "1":
            atm.check_balance()
        elif choice == "2":
            amount = float(input("Enter amount to deposit: "))
            atm.deposit(amount)
        elif choice == "3":
            amount = float(input("Enter amount to withdraw: "))
            atm.withdraw(amount)
        elif choice == "4":
            atm.change_pin()
        elif choice == "5":
            print("Thank you for using ATM. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")
```

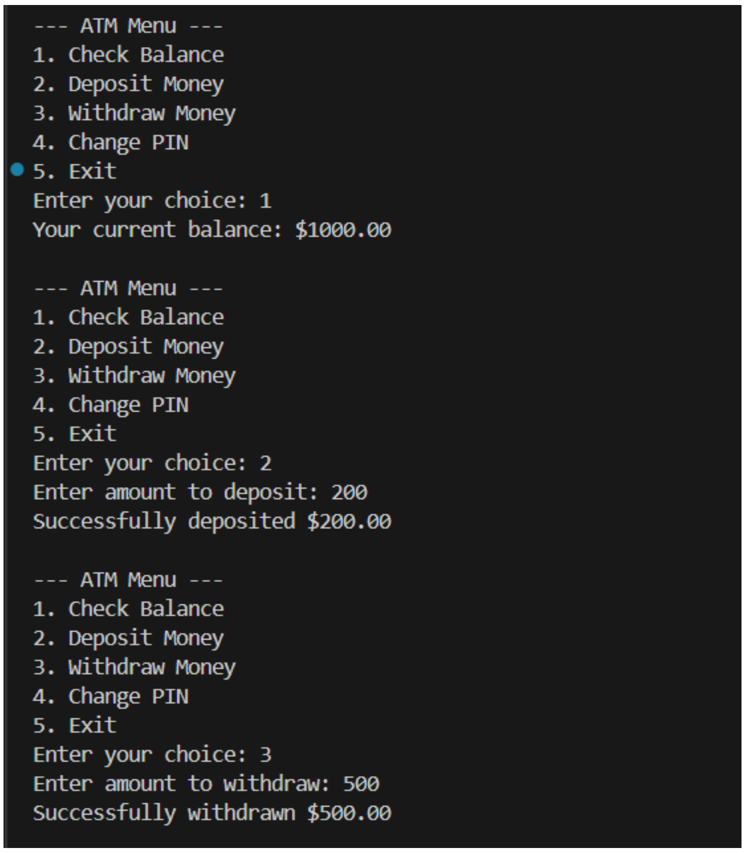
• Correct option handling.

Each menu option performs the correct action:

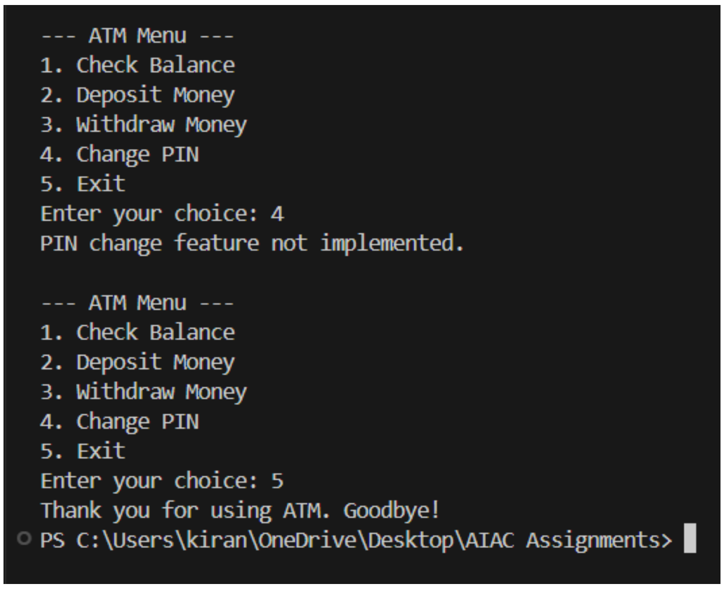
- Balance is displayed correctly.
- Deposits increase the balance.
- Withdrawals decrease the balance only if funds are sufficient.

- Invalid options are handled safely using an error message.

- **Output verification.**



```
--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Change PIN
5. Exit
Enter your choice: 1
Your current balance: $1000.00

--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Change PIN
5. Exit
Enter your choice: 2
Enter amount to deposit: 200
Successfully deposited $200.00

--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Change PIN
5. Exit
Enter your choice: 3
Enter amount to withdraw: 500
Successfully withdrawn $500.00
```



```
--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Change PIN
5. Exit
Enter your choice: 4
PIN change feature not implemented.

--- ATM Menu ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Change PIN
5. Exit
Enter your choice: 5
Thank you for using ATM. Goodbye!
PS C:\Users\kiran\OneDrive\Desktop\AIAC Assignments>
```