

# AI Assisted Coding

## Assignment 6.5

Name: T.Sriharshitha

Hall ticket no: 2303A51261

Batch no: 19

### Task 1: AI-Based Code Completion for Conditional Eligibility Check Prompt:

Generate Python code to check voting eligibility based on age and citizenship.

#### Code & Output:

The screenshot shows a code editor with two windows. The left window is titled 'Assignment\_6.5.py' and contains the following Python code:1 # Task 1: AI-Based Code Completion for Conditional Eligibility Check
2 # Generate Python code to check voting eligibility based on age and
3 # citizenship.
4 age = int(input("Enter your age: "))
5 citizenship = input("Are you a citizen? (yes/no): ").lower()
6
7 if age >= 18 and citizenship == "yes":
8 print("You are eligible to vote.")
9 else:
10 print("You are not eligible to vote.")The right window is titled 'Code' and shows the output of the code execution in a terminal window:Are you a citizen? (yes/no): yes
Are you a citizen? (yes/no): yes
You are not eligible to vote.
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment\_6.5.py"
Enter your age: 18
Are you a citizen? (yes/no): no
You are not eligible to vote.
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment\_6.5.py"
Enter your age: 18
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS E:\3rd Year\2nd Sem\AI Assisted Coding>

#### Explanation:

The AI-generated code uses conditional statements to check voting eligibility. It verifies whether the age is 18 or above and whether the user is a citizen. Both conditions must be true for eligibility. This demonstrates correct use of conditional logic generated through Albased code completion.

### Task 2: AI-Based Code Completion for Loop-Based String Processing

#### Prompt:

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

#### Code & Output:

The screenshot shows a code editor window titled "Assignment\_6.5.py". The code is a Python script designed to count vowels and consonants in a given string. It uses a loop to iterate through each character of the input string. For each character, it checks if it is alphabetic using `char.isalpha()`. If it is, it then checks if it is a vowel by comparing it against the string "aeiouAEIOU". If it is a vowel, the `vowel\_count` variable is incremented by 1. If it is not a vowel, the `consonant\_count` variable is incremented by 1. Finally, the script prints the total count of vowels and consonants. The terminal output shows the script being run with the input "Hello world", resulting in "Vowels: 3, Consonants: 7".

```
11 "Task 2: AI-Based Code Completion for Loop-Based String Processing"
12 # Generate Python code to count vowels and consonants in a string using a loop
13 input_string = input("Enter a string: ")
14 vowel_count = 0
15 consonant_count = 0
16 vowels = "aeiouAEIOU"
17 for char in input_string:
18     if char.isalpha():
19         if char in vowels:
20             vowel_count += 1
21         else:
22             consonant_count += 1
23 print(f"Vowels: {vowel_count}, Consonants: {consonant_count}")
24
```

### Explanation:

The AI-generated code processes the input string using a loop. Each character is checked to determine whether it is a vowel or a consonant. Alphabetic characters are counted correctly, while non-letter characters are ignored. The output verifies that the logic works as expected.

### Task 3: AI-Assisted Code Completion Reflection Task Prompt:

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

### Code & Output:

```
Assignment_6.5.py
Assignment_6.5 > Assignment_6.5.py > ...
24
25     # Task 3: AI-Assisted Code Completion Reflection Task"
26     # Generate a Python program for a library management system using classes,
27     # loops, and conditional statements.
28
29     class Book:
30         def __init__(self, title, author):
31             self.title = title
32             self.author = author
33             self.is_borrowed = False
34
35         def borrow(self):
36             if not self.is_borrowed:
37                 self.is_borrowed = True
38             return True
39
40         def return_book(self):
41             if self.is_borrowed:
42                 self.is_borrowed = False
43             return True
44
45     class Library:
46         def __init__(self):
47             self.books = []
48
49         def add_book(self, book):
50             self.books.append(book)
51
52         def display_books(self):
53             for idx, book in enumerate(self.books):
54                 status = "borrowed" if book.is_borrowed else "Available"
55                 print(f"[{idx + 1}]. {book.title} by {book.author} - {status}")
56
57         def borrow_book(self, index):
58             if 0 <= index < len(self.books):
59                 if self.books[index].borrow():
60                     print(f"You have borrowed '{self.books[index].title}'.")
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
```

PS E:\3rd Year\2nd Sem\AI Assisted coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment\_6.5\Assignment\_6.5.py"

1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 2  
Enter the book index to borrow: 3  
You have borrowed 'The Great Gatsby'.

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 1  
1. 1984 by George Orwell - Available  
2. To Kill a Mockingbird by Harper Lee - Available  
3. The Great Gatsby by F. Scott Fitzgerald - Borrowed

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 3  
Enter the book index to return: 3  
You have returned 'The Great Gatsby'.

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 1  
1. 1984 by George Orwell - Available  
2. To Kill a Mockingbird by Harper Lee - Available  
3. 1984 by George Orwell - Available  
4. To Kill a Mockingbird by Harper Lee - Available  
5. The Great Gatsby by F. Scott Fitzgerald - Available

```
Assignment_6.5.py
Assignment_6.5 > Assignment_6.5.py > main
44     class Library:
45
46         def borrow_book(self, index):
47             if index < 0 or index >= len(self.books):
48                 print("Invalid book index.")
49             else:
50                 if self.books[index].is_borrowed:
51                     print(f"{self.books[index].title} is already borrowed.")
52                 else:
53                     self.books[index].borrow()
54                     print(f"You have borrowed '{self.books[index].title}'.")
55
56         def return_book(self, index):
57             if 0 <= index < len(self.books):
58                 if self.books[index].return_book():
59                     print(f"You have returned '{self.books[index].title}'.")
60                 else:
61                     print(f"{self.books[index].title} was not borrowed.")
62             else:
63                 print("Invalid book index.")
64
65     def main():
66         library = Library()
67         library.add_book(Book("1984", "George Orwell"))
68         library.add_book(Book("To Kill a Mockingbird", "Harper Lee"))
69         library.add_book(Book("The Great Gatsby", "F. Scott Fitzgerald"))
70
71         while True:
72             print("\nLibrary Menu:")
73             print("1. Display Books")
74             print("2. Borrow Book")
75             print("3. Return Book")
76             print("4. Exit")
77             choice = input("Enter your choice: ")
78
79             if choice == '1':
80                 library.display_books()
81             elif choice == '2':
82                 index = int(input("Enter the book index to borrow: ")) - 1
83                 library.borrow_book(index)
84             elif choice == '3':
85                 index = int(input("Enter the book index to return: ")) - 1
86                 library.return_book(index)
87
88
89
90
91
92
93
94
```

PS E:\3rd Year\2nd Sem\AI Assisted coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment\_6.5\Assignment\_6.5.py"

Enter the book index to borrow: 3  
You have borrowed 'The Great Gatsby'.

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 1  
1. 1984 by George Orwell - Available  
2. To Kill a Mockingbird by Harper Lee - Available  
3. 1984 by George Orwell - Available  
4. To Kill a Mockingbird by Harper Lee - Available  
5. The Great Gatsby by F. Scott Fitzgerald - Available

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 3  
Enter the book index to return: 3  
You have returned 'The Great Gatsby'.

Library Menu:  
1. Display Books  
2. Borrow Book  
3. Return Book  
4. Exit

Enter your choice: 1  
1. 1984 by George Orwell - Available  
2. To Kill a Mockingbird by Harper Lee - Available  
3. 1984 by George Orwell - Available  
4. To Kill a Mockingbird by Harper Lee - Available  
5. The Great Gatsby by F. Scott Fitzgerald - Available

The screenshot shows a code editor with two tabs: 'Assignment\_6.5.py' and 'Code'. The 'Assignment\_6.5.py' tab contains Python code for a library system. The 'Code' tab shows the terminal output of running the program.

```
Assignment_6.5 > Assignment_6.5.py > main
  def main():
    library = Library()
    library.add_book("1984", "George Orwell")
    library.add_book("To Kill a Mockingbird", "Harper Lee")
    library.add_book("The Great Gatsby", "F. Scott Fitzgerald")

    while True:
        print("\nLibrary Menu:")
        print("1. Display Books")
        print("2. Borrow Book")
        print("3. Return Book")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == '1':
            library.display_books()
        elif choice == '2':
            index = int(input("Enter the book index to borrow: ")) - 1
            library.borrow_book(index)
        elif choice == '3':
            index = int(input("Enter the book index to return: ")) - 1
            library.return_book(index)
        elif choice == '4':
            print("Exiting the library system.")
            break
        else:
            print("Invalid choice. Please try again.")

    if __name__ == "__main__":
        main()

PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment_6.5\Assignment_6.5.py"
1. Display Books
2. Borrow Book
3. Return Book
4. Exit
Enter your choice: 3
Enter the book index to return: 3
You have returned 'The Great Gatsby'.

Library Menu:
1. Display Books
2. Borrow Book
3. Return Book
4. Exit
Enter your choice: 1
1. 1984 by George Orwell - Available
2. To Kill a Mockingbird by Harper Lee - Available
1. 1984 by George Orwell - Available
2. To Kill a Mockingbird by Harper Lee - Available
3. The Great Gatsby by F. Scott Fitzgerald - Available

Library Menu:
1. Display Books
2. Borrow Book
3. Return Book
4. Exit
Enter your choice: 4
Exiting the library system.
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

### Explanation:

The AI-generated program uses a class to represent a library and includes loops and conditional statements for menu-driven interaction. The loop allows continuous user input, and conditionals control program flow. The program correctly demonstrates AI-assisted use of object-oriented programming concepts.

### Reflection on AI-Assisted Coding:

The AI tool generated a complete and functional program quickly. While the logic is correct, the code can be further improved with input validation and advanced features. This task shows that AI is useful for speeding up development but still requires human review and optimization.

## Task 4: AI-Assisted Code Completion for Class-Based Attendance System

### Prompt:

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

### Code & Output:

```

102
103 "Task 4: AI-Assisted Code Completion for Class-Based Attendance System"
104 # Generate a Python class to mark and display student attendance using loops.
105 class AttendanceSystem:
106     def __init__(self):
107         self.attendance = {}
108
109     def mark_attendance(self, student_name):
110         self.attendance[student_name] = "Present"
111
112     def display_attendance(self):
113         print("Attendance Record:")
114         for student, status in self.attendance.items():
115             print(f"{student}: {status}")
116
117     def main():
118         attendance_system = AttendanceSystem()
119         while True:
120             name = input("Enter student name to mark attendance (or 'exit' to finish): ")
121             if name.lower() == 'exit':
122                 break
123             attendance_system.mark_attendance(name)
124
125 if __name__ == "__main__":
126     main()

```

### Explanation:

The AI-generated attendance system uses a class to store attendance data. A loop is used to take multiple student entries, and another loop displays the attendance records. The code works correctly and demonstrates class-based AI code completion.

## Task 5: AI-Based Code Completion for Conditional Menu Navigation

### Prompt:

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

### Code & Output:

```

120
121 "Task 5: AI-Based Code Completion for Conditional Menu Navigation"
122 # Generate a Python program using loops and conditionals to simulate an ATM
123 menu.
124 balance = 1000.0
125 while True:
126     print("\nATM Menu:")
127     print("1. Check Balance")
128     print("2. Deposit Money")
129     print("3. Withdraw Money")
130     print("4. Exit")
131     choice = input("Enter your choice: ")
132
133     if choice == '1':
134         print(f"Your current balance is: ${balance:.2f}")
135     elif choice == '2':
136         amount = float(input("Enter amount to deposit: "))
137         if amount > 0:
138             balance += amount
139             print(f"${amount:.2f} deposited successfully.")
140             print(f"Your current balance is: ${balance:.2f}")
141         else:
142             print("Invalid amount. Please enter a positive value.")
143     elif choice == '3':
144         amount = float(input("Enter amount to withdraw: "))
145         if 0 < amount <= balance:
146             balance -= amount
147             print(f"${amount:.2f} withdrawn successfully.")
148             print(f"Your current balance is: ${balance:.2f}")
149         else:
150             print("Invalid amount or insufficient balance.")
151     elif choice == '4':
152         print("Exiting the ATM. Thank you!")
153         break
154     else:
155         print("Invalid choice. Please try again.")

```

**Explanation:**

The AI-generated ATM program uses a loop to display the menu repeatedly and conditional statements to handle user choices. The logic correctly updates the balance and prevents invalid withdrawals. This task demonstrates effective AI-based code completion for menudriven programs.

**Final Conclusion:**

This experiment shows how AI-based code completion tools can generate useful Python code involving classes, loops, and conditionals. While AI speeds up development, developers must still review logic, handle edge cases, and ensure ethical and responsible use of Algenerated code.