

# AI Assisted Coding

## Assignment 6.5

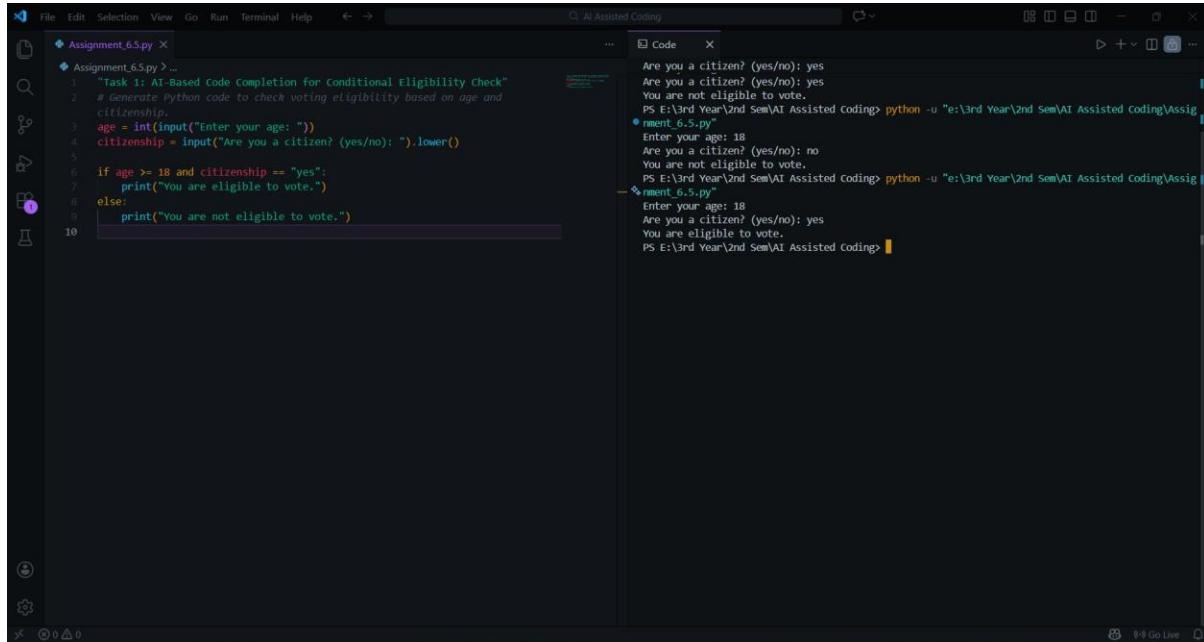
Name: ch. koushik  
Hall ticket no: 2303a51938  
Batch no: 19

### 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 interface with two panes. The left pane displays a Python script named 'Assignment\_6.5.py' containing the following code:`Assignment_6.5.py > ...
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 pane shows the terminal output of running the script with different inputs:`PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment_6.5.py"
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:

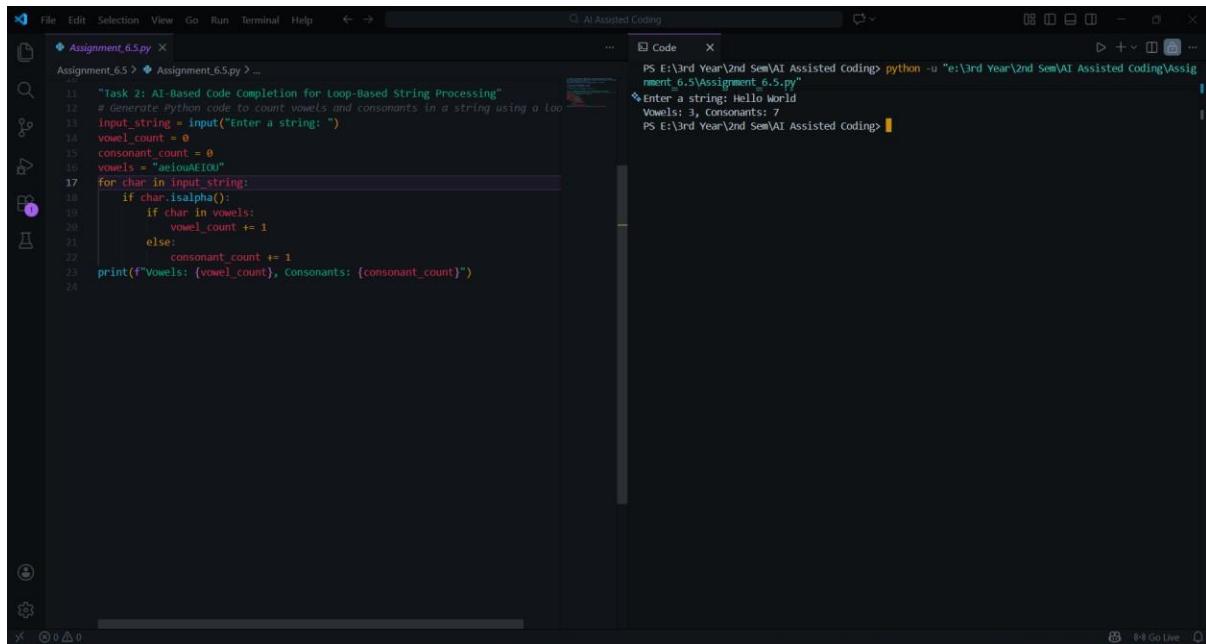
The AI-generated code uses conditional statements to check voting eligibility. It verifies whether the age is 18 or above and whether the user is a citizen. Both conditions must be true for eligibility. This demonstrates correct use of conditional logic generated through AI-based code completion.

### 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 interface with two panes. The left pane displays a Python script named `Assignment_6.5.py`. The code is as follows:

```
Assignment_6.5 > Assignment_6.5.py > ...
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
```

The right pane shows the terminal output of running the script:

```
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 a string: Hello World
Vowels: 3, Consonants: 7
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

## 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 X
Assignment_6.5 > Assignment_6.5.py > ...
25  "Task 3: AI-Assisted Code Completion Reflection Task"
26  # Generate a Python program for a library management system using classes,
27  # logs, and conditional statements.
28  class Book:
29      def __init__(self, title, author):
30          self.title = title
31          self.author = author
32          self.is_borrowed = False
33
34      def borrow(self):
35          if not self.is_borrowed:
36              self.is_borrowed = True
37              return True
38          return False
39
40      def return_book(self):
41          if self.is_borrowed:
42              self.is_borrowed = False
43              return True
44          return False
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  
1. 1984 by George Orwell - Available  
2. To Kill a Mockingbird by Harper Lee - Available  
3. The Great Gatsby by F. Scott Fitzgerald - Available

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

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  
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: 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

The screenshot shows a code editor interface with two panes. The left pane displays Python code for a library system, and the right pane shows the AI-generated output. The code defines a `Library` class with methods for adding books, displaying them, borrowing, and returning. A menu loop handles user input for these operations. The AI-generated output shows the program running, displaying a library menu and allowing the user to borrow a book.

```
Assignment_6.5.py X
Assignment_6.5 > Assignment_6.5.py > main
73     def main():
74         library = Library()
75         library.add_book(Book("1984", "George Orwell"))
76         library.add_book(Book("To Kill a Mockingbird", "Harper Lee"))
77         library.add_book(Book("The Great Gatsby", "F. Scott Fitzgerald"))
78
79     while True:
80         print("\nLibrary Menu:")
81         print("1. Display Books")
82         print("2. Borrow Book")
83         print("3. Return Book")
84         print("4. Exit")
85         choice = input("Enter your choice: ")
86
87         if choice == '1':
88             library.display_books()
89         elif choice == '2':
90             index = int(input("Enter the book index to borrow: ")) - 1
91             library.borrow_book(index)
92         elif choice == '3':
93             index = int(input("Enter the book index to return: ")) - 1
94             library.return_book(index)
95         elif choice == '4':
96             print("Exiting the library system.")
97             break
98         else:
99             print("Invalid choice. Please try again.")
100    if __name__ == "__main__":
101        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:

```

Assignment_6.5.py X
Assignment_6.5 > Assignment_6.5.py > ...
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 fin")
121             if name.lower() == "exit":
122                 break
123             attendance_system.mark_attendance(name)
124         attendance_system.display_attendance()
125
126 if __name__ == "__main__":
127     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:

```

Assignment_6.5.py X
Assignment_6.5 > Assignment_6.5.py > ...
127 "Task 5: AI-Based Code Completion for Conditional Menu Navigation"
128 # Generate a Python program using Loops and conditionals to simulate an ATM
129 menu.
130 balance = 1000.0
131 while True:
132     print("\nATM Menu:")
133     print("1. Check Balance")
134     print("2. Deposit Money")
135     print("3. Withdraw Money")
136     print("4. Exit")
137     choice = input("Enter your choice: ")
138
139     if choice == '1':
140         print(f"Your current balance is: ${balance:.2f}")
141     elif choice == '2':
142         amount = float(input("Enter amount to deposit: "))
143         if amount > 0:
144             balance += amount
145             print(f"${amount:.2f} deposited successfully.")
146         else:
147             print("Invalid amount. Please enter a positive value.")
148     elif choice == '3':
149         amount = float(input("Enter amount to withdraw: "))
150         if 0 < amount <= balance:
151             balance -= amount
152             print(f"${amount:.2f} withdrawn successfully.")
153             print(f"Your current balance is: ${balance:.2f}")
154         else:
155             print("Invalid amount or insufficient balance.")
156     elif choice == '4':
157         print("Exiting the ATM. Thank you!")
158         break
159     else:
160         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 menu-driven 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 AI-generated code.