

AI Assisted coding

Assignment-6.5

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.
- Correct eligibility decisions.
- Explanation of conditions.

The screenshot shows a code editor interface with the following details:

- Title Bar:** assign-6.5.py — ai assistedcoding
- Code Editor:** The file `assign-6.5.py` contains the following Python code:

```
1  #Generate Python code to check voting eligibility based on age and citizenship."
2  def check_voting_eligibility(age, is_citizen):
3      if age >= 18 and is_citizen:
4          return "Eligible to vote"
5      else:
6          return "Not eligible to vote"
7  # Example usage
8  age = 20
9  is_citizen = True
10 print(check_voting_eligibility(age, is_citizen))  # Output: Eligible to vote
11 age = 16
12 is_citizen = True
13 print(check_voting_eligibility(age, is_citizen))  # Output: Not eligible to vote
14 age = 25
15 is_citizen = False
16 print(check_voting_eligibility(age, is_citizen))  # Output: Not eligible to vote
17
18 |
```
- Terminal:** The terminal window shows the execution of the script and its output:

```
nithyadugyala@Nithyas-MacBook-Air-2 ~ % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
Eligible to vote
Not eligible to vote
Not eligible to vote
nithyadugyala@Nithyas-MacBook-Air-2 ~ %
```
- Right Panel:**
 - Build with Agent:** A section with a speech bubble icon and the text: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
 - Describe what to build:** A text input field with the placeholder "Describe what to build" and a "Go Live" button.
- Bottom Status Bar:** Screen Reader Optimized, Line 18, Col 1, Spaces: 4, UTF-8, LF, Python 3.14.2, Go Live, and a refresh icon.

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.
- Correct counts.
- Output verification.

The screenshot shows a code editor interface with the following details:

- Title Bar:** assign-6.5.py — ai assistedcoding
- Left Sidebar:** Includes icons for Welcome, Search, Open, Save, Undo, Redo, Find, Replace, Copy, Paste, and Settings.
- Central Area:** A code editor window titled "assign-6.5.py > ...". It contains the following Python code:

```
19  #Generate Python code to count vowels and consonants in a string
20  def count_vowels_and_consonants(input_string):
21      vowels = "aeiouAEIOU"
22      vowel_count = 0
23      consonant_count = 0
24
25      for char in input_string:
26          if char.isalpha():
27              if char in vowels:
28                  vowel_count += 1
29              else:
30                  consonant_count += 1
31
32      return vowel_count, consonant_count
33 # Example usage
34 input_string = "Hello World"
35 vowels, consonants = count_vowels_and_consonants(input_string)
36 print(f"Vowels: {vowels}, Consonants: {consonants}")
37 # Output: Vowels: 3, Consonants: 7
```
- Right Sidebar:**
 - Build with Agent:** A section with a speech bubble icon and the text "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
 - Terminal:** Shows the command run and its output:

```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
● nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
Vowels: 3, Consonants: 7
❖ nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```
 - Bottom Status Bar:** Screen Reader Optimized, Line 41, Column 1, Spaces: 4, UTF-8, LF, Python 3.14.2, Go Live, and a bell icon.

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.

The screenshot shows a code editor window titled "assign-6.5.py — ai assistedcoding". The code is a Python script for a library management system. It defines two classes: "Book" and "Library". The "Book" class has methods for initializing a book with title and author, borrowing it (which changes its availability status), and returning it. The "Library" class has methods for adding books to the collection and displaying them. The code is annotated with AI-generated comments and suggestions, particularly in the "borrow" and "display_books" methods. A sidebar on the right is titled "Build with Agent" and includes a note about AI responses being inaccurate and a link to "Generate Agent Instructions". A bottom right panel shows a summary of the file and a "Describe what to bu" input field.

```
40 #Generate a Python program for a library management system using classes, loops, and cond
41 class Book:
42     def __init__(self, title, author):
43         self.title = title
44         self.author = author
45         self.is_available = True
46
47     def borrow(self):
48         if self.is_available:
49             self.is_available = False
50             return f"You have borrowed '{self.title}' by {self.author}."
51         else:
52             return f"Sorry, '{self.title}' is currently not available."
53
54     def return_book(self):
55         self.is_available = True
56         return f"You have returned '{self.title}'. Thank you!"
57
58 class Library:
59     def __init__(self):
60         self.books = []
61
62     def add_book(self, book):
63         self.books.append(book)
64         return f"Added '{book.title}' by {book.author} to the library."
65
66     def display_books(self):
67         for book in self.books:
68             status = "Available" if book.is_available else "Not Available"
69             print(f"'{book.title}' by {book.author} - {status}")
```

```

57 class Library:
58     def __init__(self):
59         self.books = []
60
61     def add_book(self, book):
62         self.books.append(book)
63         return f"Added '{book.title}' by {book.author} to the library."
64
65     def display_books(self):
66         for book in self.books:
67             status = "Available" if book.is_available else "Not Available"
68             print(f"'{book.title}' by {book.author} - {status}")
69
70 # Example usage
71 library = Library()
72 book1 = Book("1984", "George Orwell")
73 book2 = Book("To Kill a Mockingbird", "Harper Lee")
74 print(library.add_book(book1))
75 print(library.add_book(book2))
76 library.display_books()
77 print(book1.borrow())
78 library.display_books()
79 print(book1.return_book())
80 library.display_books()

```

The screenshot shows a code editor window with a Python script named `assign-6.5.py`. The script defines a `Library` class with methods for adding books and displaying them. An AI-assisted coding interface is overlaid on the right, titled "Build with Agent". It includes a message stating "AI responses may be inaccurate." and a button to "Generate Agent Instructions to onboard AI onto your codebase". The terminal below the code shows the execution of the script and its output.

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.

```

80     #"Generate a Python class to mark and display student attendance using loops."
81     class Student:
82         def __init__(self, name):
83             self.name = name
84             self.attendance = []
85
86         def mark_attendance(self, status):
87             self.attendance.append(status)
88
89         def display_attendance(self):
90             print(f"Attendance for {self.name}:")
91             for i, status in enumerate(self.attendance, 1):
92                 print(f"Day {i}: {status}")
93
94     # Example usage
95     student = Student("Alice")
96     attendance_statuses = ["Present", "Absent", "Present", "Present", "Absent"]
97     for status in attendance_statuses:
98         student.mark_attendance(status)
99     student.display_attendance()
100
101 # Output:
102 # Attendance for Alice:
103 # Day 1: Present
104 # Day 2: Absent
105 # Day 3: Present
106 # Day 4: Present
107

```

The screenshot shows a VS Code interface with the following details:

- File:** assign-6.5.py
- Code Content:**

```

80     #"Generate a Python class to mark and display student attendance using loops."
81     class Student:
82         def __init__(self, name):
83             self.name = name
84             self.attendance = []
85
86         def mark_attendance(self, status):
87             self.attendance.append(status)
88
89         def display_attendance(self):
90             print(f"Attendance for {self.name}:")
91             for i, status in enumerate(self.attendance, 1):
92                 print(f"Day {i}: {status}")
93
94     # Example usage
95     student = Student("Alice")
96     attendance_statuses = ["Present", "Absent", "Present", "Present", "Absent"]
97     for status in attendance_statuses:
98         student.mark_attendance(status)
99     student.display_attendance()
100
101 # Output:
102 # Attendance for Alice:
103 # Day 1: Present
104 # Day 2: Absent
105 # Day 3: Present
106 # Day 4: Present
107

```
- Terminal Output:**

```

/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
● nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
Attendance for Alice:
Day 1: Present
Day 2: Absent
Day 3: Present
Day 4: Present
Day 5: Absent
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %

```
- AI Sidebar:** "Build with Agent" with a note: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
- Bottom Status Bar:** Screen Reader Optimized, Ln 79, Col 28, Spaces: 4, UTF-8, LF, Python, 3.14.2, Go Live, etc.

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

The screenshot shows a code editor window with the following Python code:

```
107 #Generate a Python program using loops and conditionals to simulate an ATM menu."
108 def atm_menu():
109     balance = 1000 # Initial balance
110     while True:
111         print("\nATM Menu:")
112         print("1. Check Balance")
113         print("2. Deposit Money")
114         print("3. Withdraw Money")
115         print("4. Exit")
116         choice = input("Please select an option (1-4): ")
117
118         if choice == '1':
119             print("Your current balance is: ${balance}")
120         elif choice == '2':
121             amount = float(input("Enter amount to deposit: $"))
122             if amount > 0:
123                 balance += amount
124                 print(f"${amount} deposited successfully.")
125             else:
126                 print("Invalid amount. Please enter a positive value.")
127         elif choice == '3':
128             amount = float(input("Enter amount to withdraw: $"))
129             if 0 < amount <= balance:
130                 balance -= amount
131                 print(f"${amount} withdrawn successfully.")
132             else:
133                 print("Invalid amount or insufficient funds.")
134         elif choice == '4':
135             print("Thank you for using the ATM. Goodbye!")
136             break
137         else:
138             print("Invalid selection. Please choose a valid option.")
139
140
141
142
```

The code defines a function `atm_menu()` that simulates an ATM menu. It uses a loop to keep the menu open until the user chooses to exit. The menu options are 1 (Check Balance), 2 (Deposit Money), 3 (Withdraw Money), and 4 (Exit). The code handles invalid inputs and ensures that deposits and withdrawals are valid amounts.

The screenshot shows the same Python code as above, but with additional comments and output from the terminal:

```
108 def atm_menu():
109     balance = 1000 # Initial balance
110     while True:
111         print("\nATM Menu:")
112         print("1. Check Balance")
113         print("2. Deposit Money")
114         print("3. Withdraw Money")
115         print("4. Exit")
116         choice = input("Please select an option (1-4): ")
117
118         if choice == '1':
119             print("Your current balance is: ${balance}")
120         elif choice == '2':
121             amount = float(input("Enter amount to deposit: $"))
122             if amount > 0:
123                 balance += amount
124                 print(f"${amount} deposited successfully.")
125             else:
126                 print("Invalid amount. Please enter a positive value.")
127         elif choice == '3':
128             amount = float(input("Enter amount to withdraw: $"))
129             if 0 < amount <= balance:
130                 balance -= amount
131                 print(f"${amount} withdrawn successfully.")
132             else:
133                 print("Invalid amount or insufficient funds.")
134         elif choice == '4':
135             print("Thank you for using the ATM. Goodbye!")
136             break
137         else:
138             print("Invalid selection. Please choose a valid option.")
139
140
141
142
```

The terminal output shows the execution of the script and the resulting ATM menu interaction:

```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ~ ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 1
Your current balance is: $1000
ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4):
```