

# AI Assisted coding

## Assignment-6.5

S.Rajesh

2303a52301

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 with a file named 'assign-6.5.py'. The code defines a function 'check\_voting\_eligibility' that takes 'age' and 'is\_citizen' as arguments. It returns 'Eligible to vote' if the age is 18 or older and the person is a citizen, otherwise it returns 'Not eligible to vote'. Below the function, there are example usage lines with comments showing the expected output for different inputs.

```
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
8  # Example usage
9  age = 20
10 is_citizen = True
11 print(check_voting_eligibility(age, is_citizen)) # Output: Eligible to vote
12 age = 16
13 is_citizen = True
14 print(check_voting_eligibility(age, is_citizen)) # Output: Not eligible to vote
15 age = 25
16 is_citizen = False
17 print(check_voting_eligibility(age, is_citizen)) # Output: Not eligible to vote
18
```

The terminal output shows the execution of the code, confirming the expected results:

```
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/a
i assistedcoding/assign-6.5.py"
Eligible to vote
Not eligible to vote
Not eligible to vote
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

## 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 with a Python file named `assign-6.5.py`. The code defines a function `count_vowels_and_consonants` that takes an `input_string` and returns the count of vowels and consonants. It uses a set of vowels `"aeiouAEIOU"` and increments counters for each character in the input string. An example usage is provided with the string `"Hello World"`, resulting in 3 vowels and 7 consonants.

```
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
34 # Example usage
35 input_string = "Hello World"
36 vowels, consonants = count_vowels_and_consonants(input_string)
37 print(f"Vowels: {vowels}, Consonants: {consonants}")
38 # Output: Vowels: 3, Consonants: 7
```

The terminal output shows the execution of the program:

```
/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/a
i assistedcoding/assign-6.5.py"
Vowels: 3, Consonants: 7
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

The sidebar on the right contains a section titled "Build with Agent" with the text "AI responses may be inaccurate." and a link "Generate Agent Instructions to onboard AI onto your codebase."

### 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.

← → assign-6.5.py — ai assistedcoding

Welcome assign-6.5.py Library

```
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 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}")
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

assign-6.5.py +

Describe what to bui

< 0 0 0 Screen Reader Optimized Ln 60, Col 1 Spaces: 4 UTF-8 LF Python 3.14.2 Go Live

← → assign-6.5.py — ai assistedcoding

Welcome assign-6.5.py X

```
57 class Library:
58     self.books = []
59
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()
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

assign-6.5.py +

Describe what to bui

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python + Python 3.14.2 Go Live

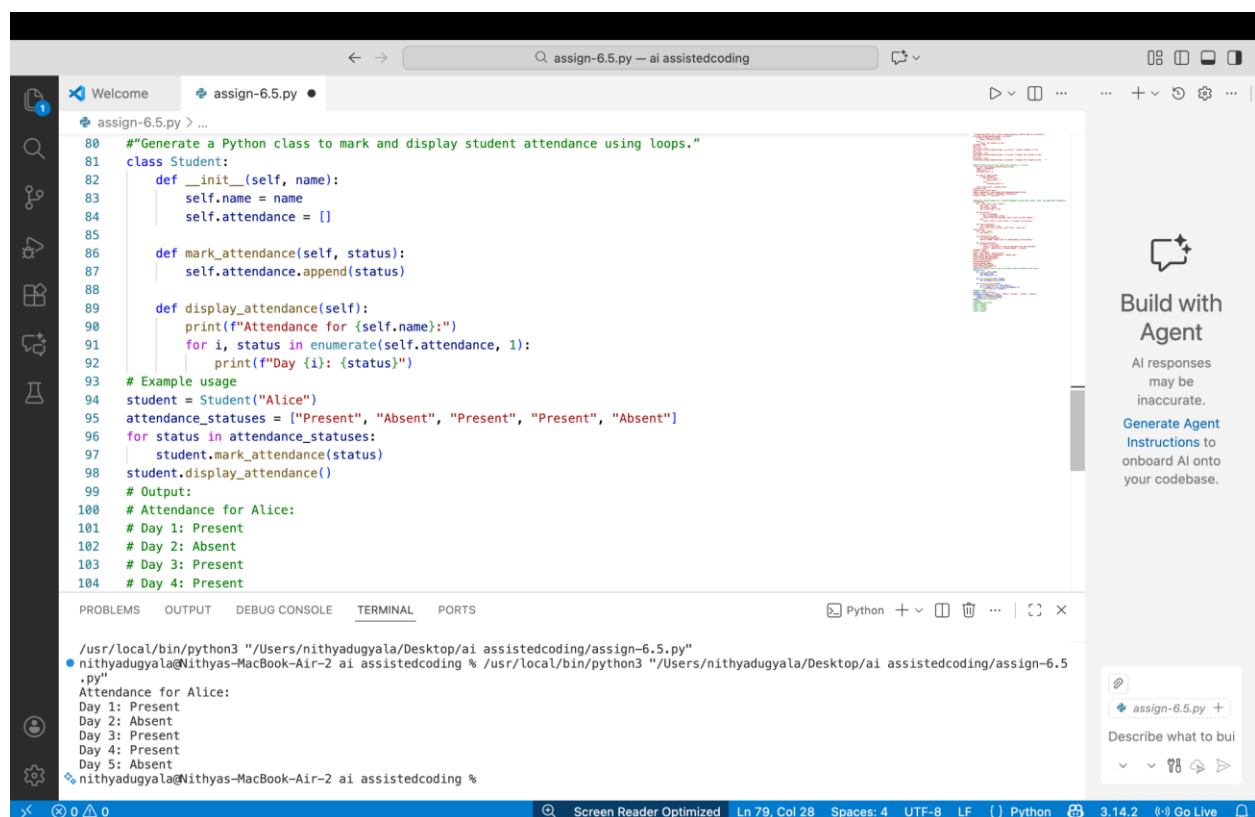
```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/a
i assistedcoding/assign-6.5.py"
Added '1984' by George Orwell to the library.
Added 'To Kill a Mockingbird' by Harper Lee to the library.
'1984' by George Orwell - Available
'To Kill a Mockingbird' by Harper Lee - Available
You have borrowed '1984' by George Orwell.
'1984' by George Orwell - Not Available
'To Kill a Mockingbird' by Harper Lee - Available
You have returned '1984'. Thank you!
'1984' by George Orwell - Available
'To Kill a Mockingbird' by Harper Lee - Available
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

## 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.



The screenshot shows a code editor with a Python file named 'assign-6.5.py'. The code defines a 'Student' class with methods for marking and displaying attendance. It includes example usage and expected output.

```
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 # Day 5: Absent
```

The terminal output shows the execution of the code, displaying the attendance for Alice over five days.

```
/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 %
```

## 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

← → assign-6.5.py — ai assistedcoding

Welcome assign-6.5.py

assign-6.5.py > atm\_menu

```
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(f"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.")
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

assign-6.5.py +

Describe what to bui

Screen Reader Optimized Ln 138, Col 70 Spaces: 4 UTF-8 LF Python 3.14.2 Go Live

← → assign-6.5.py — ai assistedcoding

Welcome assign-6.5.py

assign-6.5.py > ...

```
108 def atm_menu():
125     print(f"${amount} deposited successfully.")
126     else:
127         print("Invalid amount. Please enter a positive value.")
128     elif choice == '3':
129         amount = float(input("Enter amount to withdraw: $"))
130         if 0 < amount <= balance:
131             balance -= amount
132             print(f"${amount} withdrawn successfully.")
133         else:
134             print("Invalid amount or insufficient funds.")
135     elif choice == '4':
136         print("Thank you for using the ATM. Goodbye!")
137         break
138     else:
139         print("Invalid selection. Please choose a valid option.")
140 # Example usage
141 atm_menu()
142 # Output will vary based on user input
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

assign-6.5.py +

Describe what to bui

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python + Python +

```
/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):
```

Screen Reader Optimized Ln 142, Col 1 Spaces: 4 UTF-8 LF Python 3.14.2 Go Live