

# AI ASSISTED -CODING

## ASSIGNMENT-6.5

RUDROJU RUPA SRI

2303A51918

BATCH-30

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

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.
- Correct eligibility decisions.
- Explanation of conditions.

The screenshot shows a code editor with a file named 'ASS-6.5.py' open. The code is a Python script that takes user input for age and citizenship, checks if the user is eligible to vote, and prints the result. The script is as follows:

```
1 # Take age and citizenship as input
2 # Use conditional statements to check voting eligibility
3 # Eligible if age >= 18 and citizen is True
4 # Print eligibility result with proper message
5
6 age = int(input("Enter your age: "))
7 citizen = input("Are you a citizen? (yes/no): ").lower() == "yes"
8
9 if age >= 18 and citizen:
10     print("You are eligible to vote.")
11 else:
12     print("You are not eligible to vote.")
```

Below the code editor is a terminal window showing the execution of the script. The terminal output is as follows:

```
PS C:\Users\HP\Desktop\AI> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/AI/ASS-6.5.py
Enter your age: 21
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS C:\Users\HP\Desktop\AI>
```

The status bar at the bottom of the editor shows 'Ln 10, Col 39', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python', 'Python 3.13 (64-bit)', and 'Go Live'.

## Task Description #2(AI-Based Code Completion for Loop-Based String Processing)

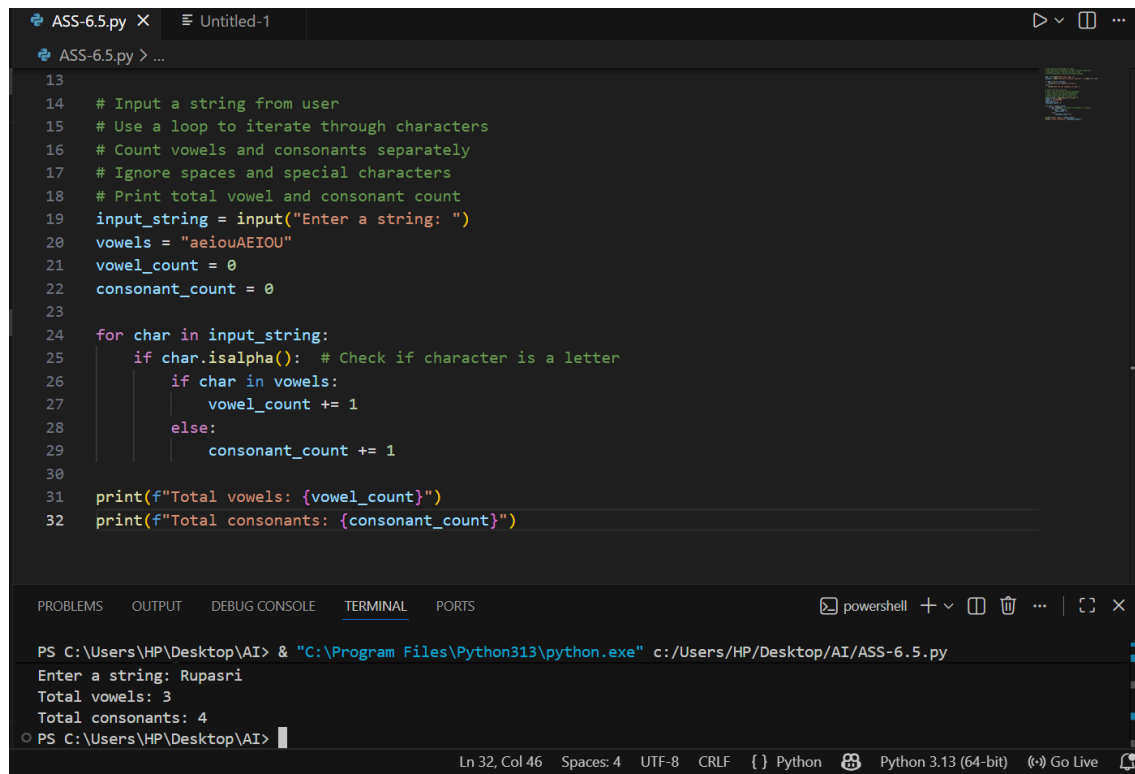
Task: Use an AI tool to process strings using loops.

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 image shows a Visual Studio Code editor window with a Python file named 'ASS-6.5.py'. The code is a script to count vowels and consonants in a user-provided string. The code is as follows:

```
13
14 # Input a string from user
15 # Use a loop to iterate through characters
16 # Count vowels and consonants separately
17 # Ignore spaces and special characters
18 # Print total vowel and consonant count
19 input_string = input("Enter a string: ")
20 vowels = "aeiouAEIOU"
21 vowel_count = 0
22 consonant_count = 0
23
24 for char in input_string:
25     if char.isalpha(): # Check if character is a letter
26         if char in vowels:
27             vowel_count += 1
28         else:
29             consonant_count += 1
30
31 print(f"Total vowels: {vowel_count}")
32 print(f"Total consonants: {consonant_count}")
```

Below the code editor is a terminal window showing the execution of the script. The prompt 'Enter a string:' is followed by the input 'Rupasri'. The output shows 'Total vowels: 3' and 'Total consonants: 4'.

```
PS C:\Users\HP\Desktop\AI> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/AI/ASS-6.5.py
Enter a string: Rupasri
Total vowels: 3
Total consonants: 4
PS C:\Users\HP\Desktop\AI>
```

### 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.



```
ASS-6.5.py > ...
66 # Student Attendance System
67 # Create a simple Attendance class
68 # Use loop to input student names and P/A status
69 # Store attendance in a dictionary
70 # Display present and absent students
71 class Attendance:
72     def __init__(self):
73         self.attendance_record = {}
74     def mark_attendance(self, name, status):
75         self.attendance_record[name] = status
76     def display_attendance(self):
77         present_students = [name for name, status in self.attendance_record.items() if status == 'P']
78         absent_students = [name for name, status in self.attendance_record.items() if status == 'A']
79         print(f"Present Students: {' '.join(present_students)}")
80         print(f"Absent Students: {' '.join(absent_students)}")
81 # Example usage
82 if __name__ == "__main__":
83     attendance = Attendance()
84     num_students = int(input("Enter the number of students: "))
85     for _ in range(num_students):
86         name = input("Enter student name: ")
87         status = input("Enter attendance status (P/A): ")
88         attendance.mark_attendance(name, status)
89     attendance.display_attendance()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\HP\Desktop\AI> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/AI/ASS-6.5.py
Enter student name: rupa
Enter attendance status (P/A): A
Enter attendance status (P/A): A
Enter student name: spurthi
Enter attendance status (P/A): P
Enter student name: hruthika
Enter attendance status (P/A): P
Present Students: spurthi, hruthika
Absent Students: rupa
```

## 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.

```
ASS-6.5.py X Untitled-1
ASS-6.5.py > generate_atm_menu > ATM > display_menu
90
91 #Task:5
92 def generate_atm_menu():
93     """
94     Function to request AI to generate a Python program that simulates an ATM menu.
95
96     Returns:
97     AI-generated code for an ATM menu with loops and conditionals.
98     """
99     class ATM:
100         def __init__(self, balance=0):
101             self.balance = balance
102
103         def display_menu(self):
104             print("ATM Menu:")
105             print("1. Check Balance")
106             print("2. Deposit")
107             print("3. Withdraw")
108             print("4. Exit")
109
110         def check_balance(self):
111             print(f"Your current balance is: ${self.balance}")
112
113         def deposit(self, amount):
114             self.balance += amount
115             print(f"${amount} deposited successfully.")
116
117         def withdraw(self, amount):
118             if amount > self.balance:
119                 print("Insufficient funds.")
120             else:
121                 self.balance -= amount
122                 print(f"${amount} withdrawn successfully.")
123
124     # Example usage
125     if __name__ == "__main__":
126         atm = ATM(1000) # Initial balance of $1000
127         while True:
128             atm.display_menu()
129             choice = input("Enter your choice: ")
130             if choice == '1':
131                 atm.check_balance()
132             elif choice == '2':
133                 amount = float(input("Enter amount to deposit: "))
134                 atm.deposit(amount)
135             elif choice == '3':
136                 amount = float(input("Enter amount to withdraw: "))
137                 atm.withdraw(amount)
138             elif choice == '4':
139                 print("Thank you for using the ATM. Goodbye!")
140                 break
141             else:
142                 print("Invalid choice. Please try again.")
143         generate_atm_menu()
```

```
ASS-6.5.py X Untitled-1
ASS-6.5.py > generate_atm_menu > ATM > display_menu
92 def generate_atm_menu():
99     class ATM:
117         def withdraw(self, amount):
118             if amount > self.balance:
119                 print("Insufficient funds.")
120             else:
121                 self.balance -= amount
122                 print(f"${amount} withdrawn successfully.")
123
124     # Example usage
125     if __name__ == "__main__":
126         atm = ATM(1000) # Initial balance of $1000
127         while True:
128             atm.display_menu()
129             choice = input("Enter your choice: ")
130             if choice == '1':
131                 atm.check_balance()
132             elif choice == '2':
133                 amount = float(input("Enter amount to deposit: "))
134                 atm.deposit(amount)
135             elif choice == '3':
136                 amount = float(input("Enter amount to withdraw: "))
137                 atm.withdraw(amount)
138             elif choice == '4':
139                 print("Thank you for using the ATM. Goodbye!")
140                 break
141             else:
142                 print("Invalid choice. Please try again.")
143         generate_atm_menu()
```

