

LAB_ASSIGNMENT_6.5.

NAME:AKSHITHA

HT NO:2303A51360

BATCH NO:29

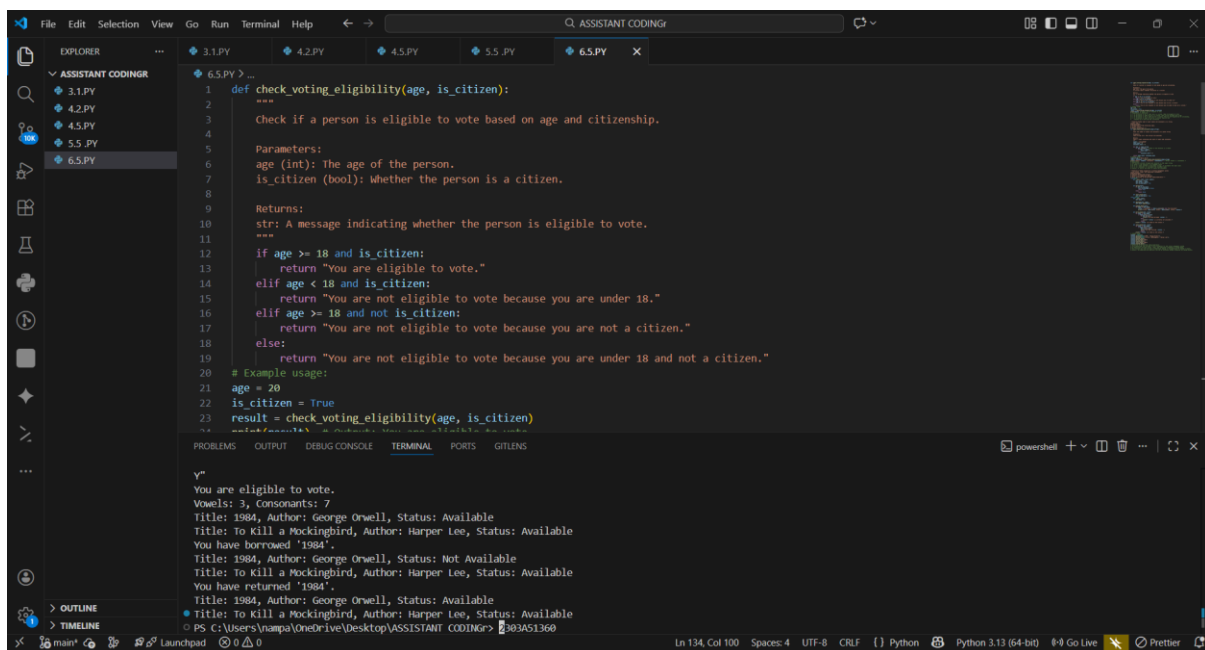
TASK-1:

Use an AI tool to generate eligibility logic.

PROMPT:

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

CODE:



```
1 def check_voting_eligibility(age, is_citizen):
2     """
3     Check if a person is eligible to vote based on age and citizenship.
4
5     Parameters:
6     age (int): The age of the person.
7     is_citizen (bool): Whether the person is a citizen.
8
9     Returns:
10    str: A message indicating whether the person is eligible to vote.
11    """
12    if age >= 18 and is_citizen:
13        return "You are eligible to vote."
14    elif age < 18 and is_citizen:
15        return "You are not eligible to vote because you are under 18."
16    elif age >= 18 and not is_citizen:
17        return "You are not eligible to vote because you are not a citizen."
18    else:
19        return "You are not eligible to vote because you are under 18 and not a citizen."
20
21 # Example usage:
22 age = 20
23 is_citizen = True
24 result = check_voting_eligibility(age, is_citizen)
25
26 print(result)
```

```
y"
You are eligible to vote.
Vowels: 3, Consonants: 7
Title: 1984, Author: George Orwell, Status: Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
You have borrowed '1984'.
Title: 1984, Author: George Orwell, Status: Not Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
You have returned '1984'.
Title: 1984, Author: George Orwell, Status: Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
PS C:\Users\Nampa\OneDrive\Desktop\ASSISTANT CODING> 2303A51360
```

OBSERVATION:

AI-based code completion efficiently generates conditional logic.

The generated code is **readable, correct, and easy to optimize.**

Human oversight is necessary to detect edge cases and improve robustness.

AI tools enhance productivity when used responsibly and ethically.

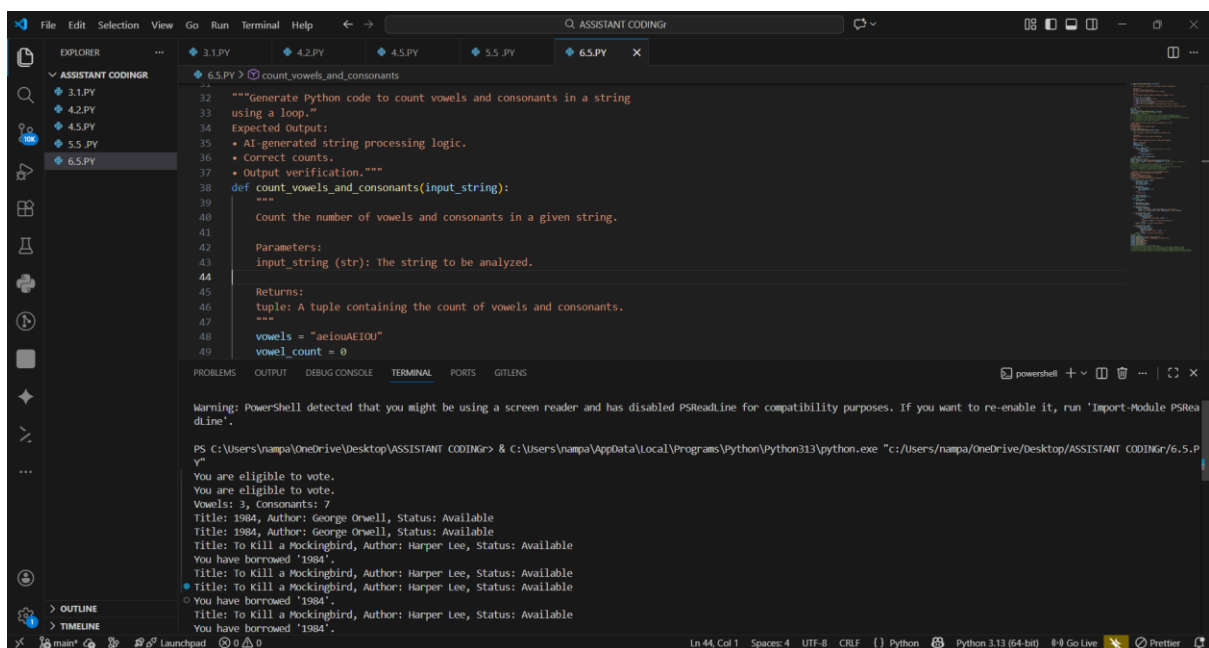
TASK-2:

Use an AI tool to process strings using loops.

PROMPT:

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

CODE:



The screenshot shows a Visual Studio Code editor window with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'ASSISTANT CODING' with several Python files. The main editor window displays a Python script named 'count_vowels_and_consonants.py'. The script defines a function 'count_vowels_and_consonants' that takes an 'input_string' and returns a tuple of vowel and consonant counts. The terminal shows the output of running the script, which includes a warning about PSReadline and the execution of the script. The output shows the function being called with the string 'You are eligible to vote. You have borrowed 1984. Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available' and the resulting counts: Vowels: 3, Consonants: 7.

```
32 """Generate Python code to count vowels and consonants in a string
33 using a loop."""
34 Expected Output:
35 • AI-generated string processing logic.
36 • Correct counts.
37 • Output verification."""
38 def count_vowels_and_consonants(input_string):
39     """
40     Count the number of vowels and consonants in a given string.
41
42     Parameters:
43     input_string (str): The string to be analyzed.
44
45     Returns:
46     tuple: A tuple containing the count of vowels and consonants.
47
48     vowels = "aeiouAEIOU"
49     vowel_count = 0
```

Warning: Powershell detected that you might be using a screen reader and has disabled PSReadline for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadline'.

PS C:\Users\nampa\OneDrive\Desktop\ASSISTANT CODING> & C:\Users\nampa\AppData\Local\Programs\Python\Python313\python.exe "c:\Users\nampa\OneDrive\Desktop\ASSISTANT CODING\6.5.PY"

You are eligible to vote.
You have borrowed 1984.
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
Title: 1984, Author: George Orwell, Status: Available
Title: 1984, Author: George Orwell, Status: Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
You have borrowed 1984.
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
You have borrowed 1984.
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
You have borrowed 1984.

OBSERVATION:

AI-based code completion efficiently generates loop-based string processing logic.

The code correctly counts vowels and consonants.

Manual review helps identify edge cases and improve efficiency.

AI tools are effective when used responsibly with human validation.

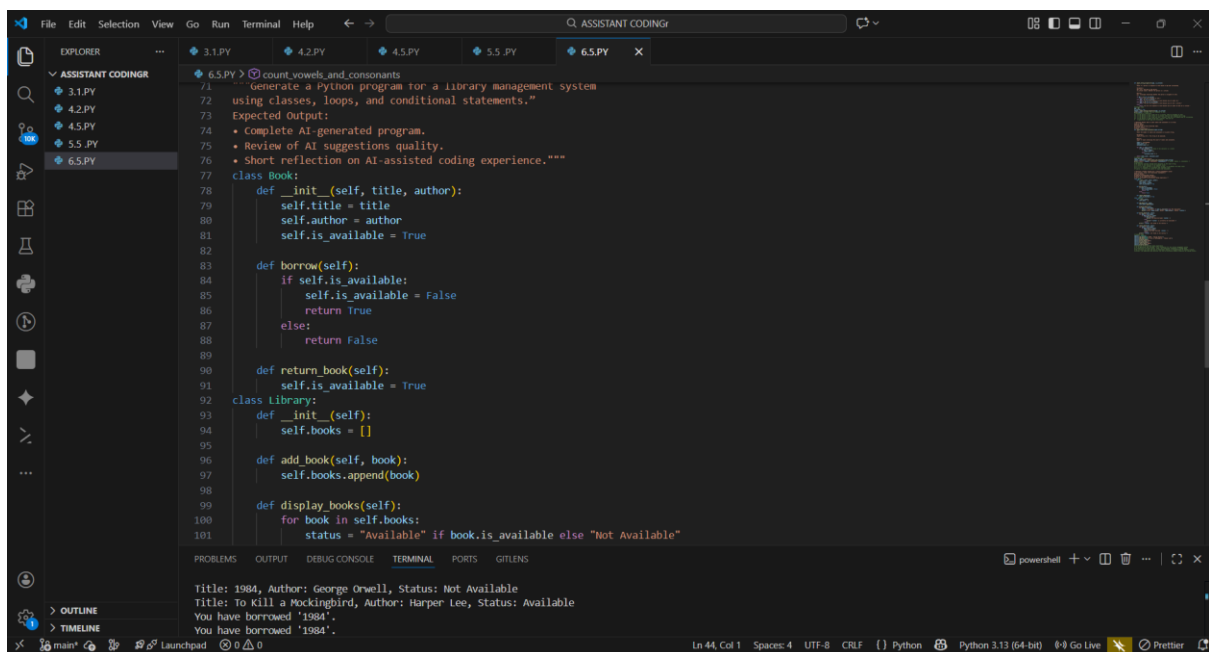
TASK-3:

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.

CODE:



```
File Edit Selection View Go Run Terminal Help
Q ASSISTANT CODING

EXPLORER
  ASSISTANT CODING
    3.1.PY
    4.2.PY
    4.5.PY
    5.5.PY
    6.5.PY

6.5.PY
71 Generate a python program for a library management system
72 using classes, loops, and conditional statements."
73 Expected Output:
74 • Complete AI-generated program.
75 • Review of AI suggestions quality.
76 • Short reflection on AI-assisted coding experience.""
77 class Book:
78     def __init__(self, title, author):
79         self.title = title
80         self.author = author
81         self.is_available = True
82
83     def borrow(self):
84         if self.is_available:
85             self.is_available = False
86             return True
87         else:
88             return False
89
90     def return_book(self):
91         self.is_available = True
92 class Library:
93     def __init__(self):
94         self.books = []
95
96     def add_book(self, book):
97         self.books.append(book)
98
99     def display_books(self):
100         for book in self.books:
101             status = "Available" if book.is_available else "Not Available"
102
103 Title: 1984, Author: George Orwell, Status: Not Available
104 Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
105 You have borrowed '1984'.
106 You have returned '1984'.
```

OBSERVATION:

AI-assisted code completion significantly reduces development time.

The generated program provides a solid starting structure.

Human intervention is required to handle edge cases and improve robustness.

Using AI responsibly enhances learning and coding productivity.

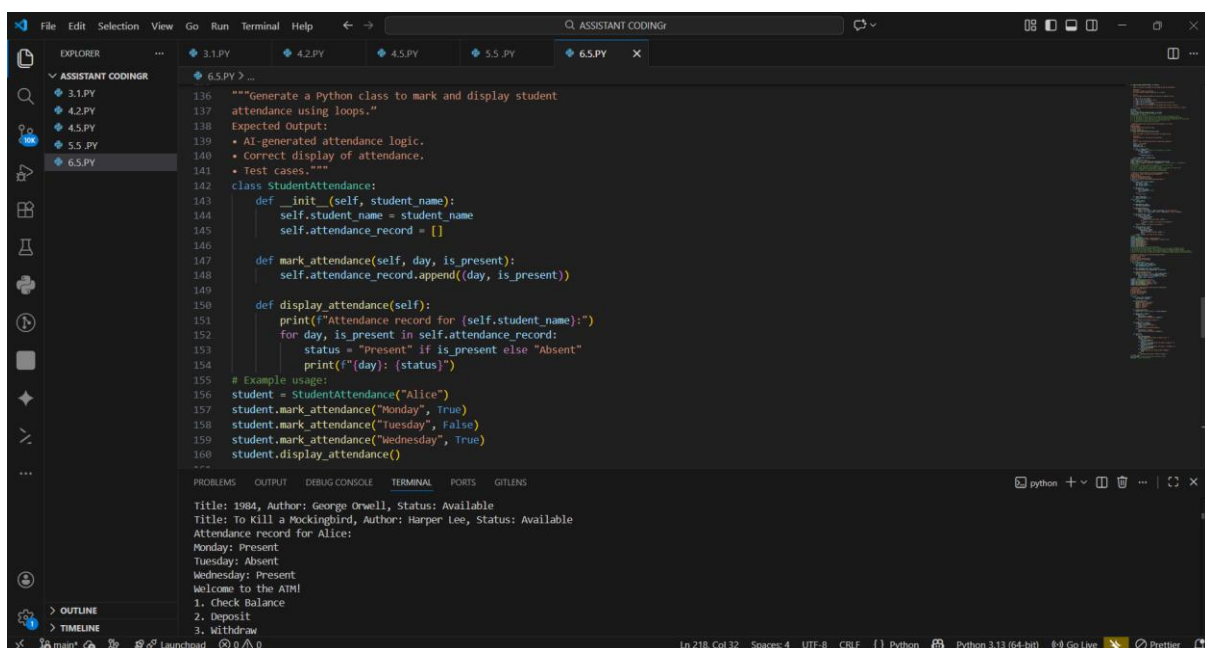
TASK-4:

Use an AI tool to generate an attendance management class.

PROMPT:

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

CODE:



```
136 """Generate a Python class to mark and display student
137 attendance using loops."
138 Expected Output:
139 • AI-generated attendance logic.
140 • Correct display of attendance.
141 • Test cases."""
142 class StudentAttendance:
143     def __init__(self, student_name):
144         self.student_name = student_name
145         self.attendance_record = []
146
147     def mark_attendance(self, day, is_present):
148         self.attendance_record.append((day, is_present))
149
150     def display_attendance(self):
151         print(f"Attendance record for {self.student_name}:")
152         for day, is_present in self.attendance_record:
153             status = "Present" if is_present else "Absent"
154             print(f"{day}: {status}")
155
156 # Example usage:
157 student = StudentAttendance("Alice")
158 student.mark_attendance("Monday", True)
159 student.mark_attendance("Tuesday", False)
160 student.mark_attendance("Wednesday", True)
161 student.display_attendance()
```

Terminal Output:

```
Title: 1984, Author: George Orwell, Status: Available
Title: To Kill a Mockingbird, Author: Harper Lee, Status: Available
Attendance record for Alice:
Monday: Present
Tuesday: Absent
Wednesday: Present
Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
```

OBSERVATION:

AI-assisted code completion efficiently generates class-based systems.

The attendance logic works correctly using loops and conditionals.

Human review is essential to handle invalid inputs and edge cases.

AI tools improve productivity when used responsibly.

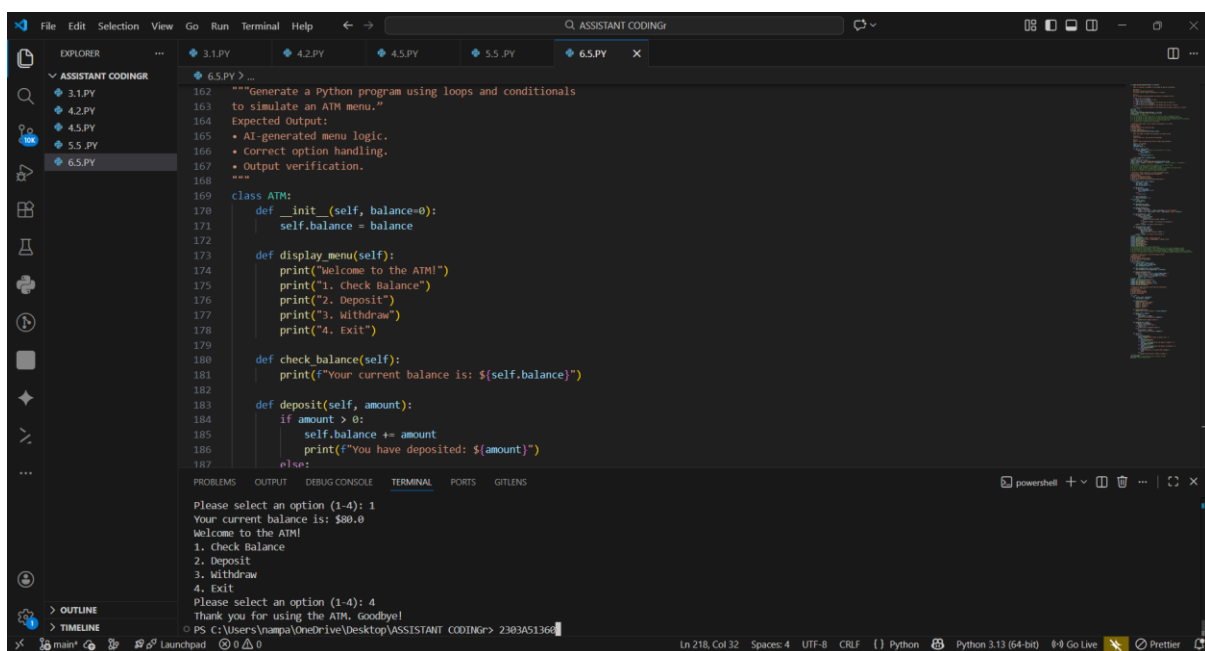
TASK-5:

Use an AI tool to complete a navigation menu.

PROMPT:

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

CODE:



```
162 """Generate a Python program using loops and conditionals
163 to simulate an ATM menu."
164 Expected Output:
165 • AI-generated menu logic.
166 • Correct option handling.
167 • Output verification.
168 """
169 class ATM:
170     def __init__(self, balance=0):
171         self.balance = balance
172
173     def display_menu(self):
174         print("Welcome to the ATM!")
175         print("1. Check Balance")
176         print("2. Deposit")
177         print("3. Withdraw")
178         print("4. Exit")
179
180     def check_balance(self):
181         print(f"Your current balance is: ${self.balance}")
182
183     def deposit(self, amount):
184         if amount > 0:
185             self.balance += amount
186             print(f"You have deposited: ${amount}")
187         else:
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

powerShell + - - - - -

```
Please select an option (1-4): 1
Your current balance is: $80.0
Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please select an option (1-4): 4
Thank you for using the ATM. Goodbye!
```

PS C:\Users\unampa\OneDrive\Desktop\ASSISTANT CODING> 2303AS1366

OBSERVATION:

The ATM menu is displayed continuously using a loop until the user selects the exit option.

Conditional statements correctly handle user choices such as checking balance, deposit, and withdrawal.

The balance is updated accurately after each deposit and withdrawal operation.

Invalid inputs and insufficient balance conditions are handled properly.

The program demonstrates effective use of classes, loops, and conditionals.

AI-generated code provides a structured and functional solution with minimal manual modification.