

AI ASSISTED CODING

LAB 6.5

2303A51733

T.PRANATHI

BATCH-11

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.

PROMPT:

#Give me a python code to check whether the person is eligible to vote or not based on their citizenship

CODE:

The screenshot shows a code editor window with a dark theme. The file is named 'lab5.py'. The code defines a function 'is_eligible_to_vote' that checks if a person is eligible to vote based on their age (18 or older) and citizenship ('yes'). It includes example usage code that prompts the user for their age and citizenship status, then prints whether they are eligible to vote.

```
lab5.py > ...
1  #Give me a python code to check whether the person is eligible to vote or not based on their
2  def is_eligible_to_vote(age, citizenship):
3      if age >= 18 and citizenship.lower() == "yes":
4          return True
5      else:
6          return False
7  # Example usage
8  age = int(input("Enter your age: "))
9  citizenship = input("Are you a citizen? (yes/no): ")
10 if is_eligible_to_vote(age, citizenship):
11     print("You are eligible to vote.")
12 else:
13     print("You are not eligible to vote.")
14
```

OUTPUT:

The screenshot shows the 'TERMINAL' tab in VS Code. It displays two separate runs of the 'lab5.py' script. In the first run, the user enters '18' for age and 'yes' for citizenship, receiving the output 'You are eligible to vote.'. In the second run, the user enters '17' for age and 'no' for citizenship, receiving the output 'You are not eligible to vote.'. The terminal also shows the command prompt 'PS C:\Users\thota\OneDrive\Desktop\AIAC>' and the path 'C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py'.

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... x
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py
Enter your age: 18
Are you a citizen? (yes/no): yes
You are eligible to vote.
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py
Enter your age: 17
Are you a citizen? (yes/no): no
You are not eligible to vote.
○ PS C:\Users\thota\OneDrive\Desktop\AIAC>
```

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:

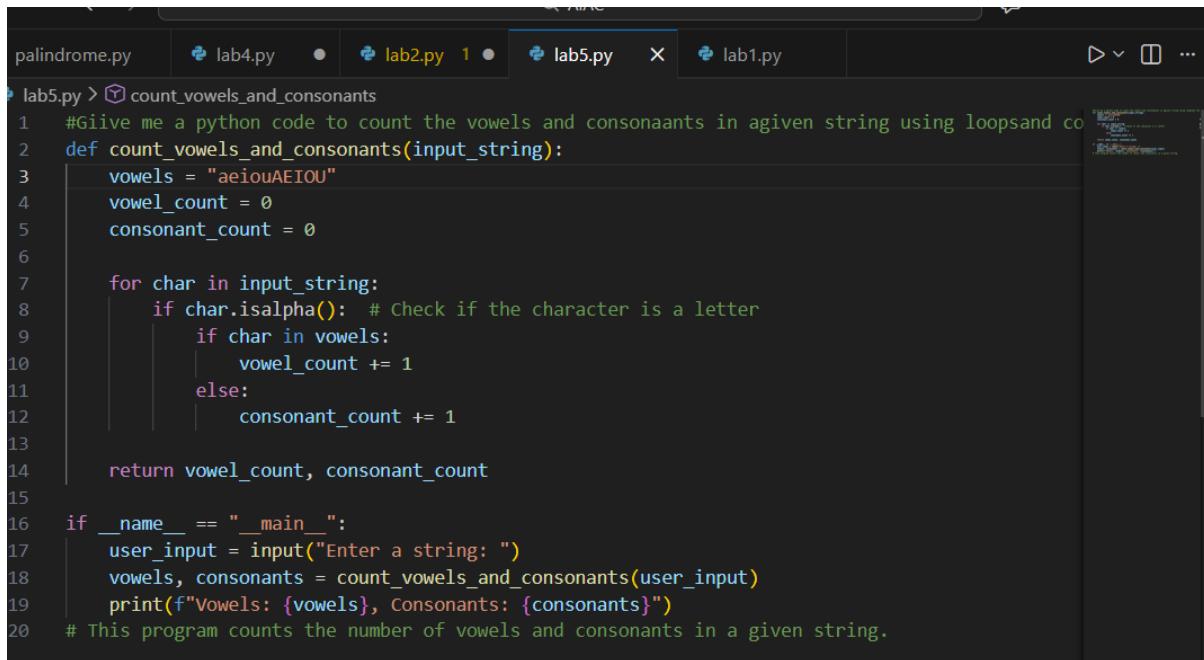
- AI-generated string processing logic.

- Correct counts.
- Output verification.

PROMPT:

#Give me a python code to count the vowels and consonants in a given string using loops and conditional statements with using input by user

CODE:

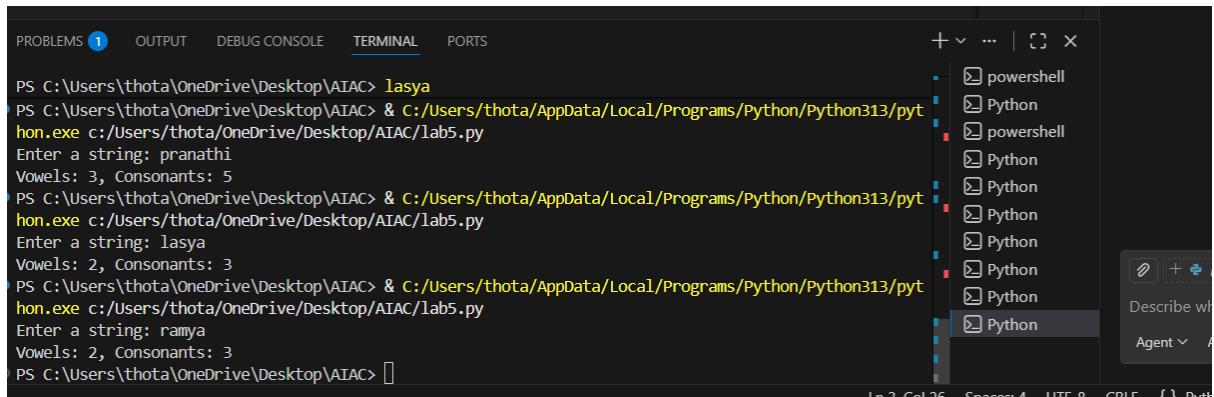


```

palindrome.py    lab4.py    lab2.py  1  lab5.py    lab1.py
lab5.py > ⚡ count_vowels_and_consonants
1 #Give me a python code to count the vowels and consonants in a given string using loops and co
2 def count_vowels_and_consonants(input_string):
3     vowels = "aeiouAEIOU"
4     vowel_count = 0
5     consonant_count = 0
6
7     for char in input_string:
8         if char.isalpha(): # Check if the character is a letter
9             if char in vowels:
10                 vowel_count += 1
11             else:
12                 consonant_count += 1
13
14     return vowel_count, consonant_count
15
16 if __name__ == "__main__":
17     user_input = input("Enter a string: ")
18     vowels, consonants = count_vowels_and_consonants(user_input)
19     print(f"Vowels: {vowels}, Consonants: {consonants}")
20 # This program counts the number of vowels and consonants in a given string.

```

OUTPUT:



```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\thota\OneDrive\Desktop\AIAC> lasya
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/Lab5.py
Enter a string: pranathi
Vowels: 3, Consonants: 5
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py
Enter a string: lasya
Vowels: 2, Consonants: 3
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py
Enter a string: ramya
Vowels: 2, Consonants: 3
PS C:\Users\thota\OneDrive\Desktop\AIAC>

```

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.

PROMPT:

#Write a python code for library management system using classes,loops and conditionl statements with input by user

CODE:

```

palindrome.py  lab4.py  lab2py 1  lab5.py 2  lab1.py
lab5.py > Book > borrow
1 #Write a python code for library management system using classes,loops and conditional statements with input by user
2 class Book:
3     def __init__(self, title, author):
4         self.title = title
5         self.author = author
6         self.is_borrowed = False
7     def borrow(self):
8         if not self.is_borrowed:
9             self.is_borrowed = True
10            return True
11        return False
12    def return_book(self):
13        if self.is_borrowed:
14            self.is_borrowed = False
15            return True
16        return False
17 class Library:
18     def __init__(self):
19         self.books = []
20     def add_book(self, book):
21         self.books.append(book)
22     def display_books(self):
23         for idx, book in enumerate(self.books):
24             status = "Borrowed" if book.is_borrowed else "Available"
25             print(f'{idx + 1}. {book.title} by {book.author} - {status}')
26     def borrow_book(self, book_index):
27         if 0 <= book_index < len(self.books):
28             if self.books[book_index].borrow():
29                 print(f"You have borrowed '{self.books[book_index].title}'")
30             else:
31                 print(f"Sorry, '{self.books[book_index].title}' is already borrowed.")
32         else:
33             print("Invalid book index.") def return_book(self, book_index):
34         if 0 <= book_index < len(self.books):
35             if self.books[book_index].return_book():
36                 print(f"You have returned '{self.books[book_index].title}'")
37             else:
38                 print(f"'{self.books[book_index].title}' was not borrowed.")
39         else:
40             print("Invalid book index.")
41 def main():
42     library = Library()
43     while True:
44         print("\nLibrary Management System")
45         print("1. Add Book")
46         print("2. Display Books")
47         print("3. Borrow Book")
48         print("4. Return Book")
49         print("5. Exit")
50         choice = input("Enter your choice: ")
51         if choice == '1':
52             title = input("Enter book title: ")
53             author = input("Enter book author: ")
54             book = Book(title, author)
55             library.add_book(book)
56             print(f"Book '{title}' added to the library.")
57         elif choice == '2':
58             library.display_books()
59         elif choice == '3':
60             library.display_books()
61             book_index = int(input("Enter the book index to borrow: ")) - 1
62             library.borrow_book(book_index)
63         elif choice == '4':
64             library.display_books()
65             book_index = int(input("Enter the book index to return: ")) - 1
66             library.return_book(book_index)
67         elif choice == '5':
68             print("Exiting the system. Goodbye!")
69             break
70         else:
71             print("Invalid choice. Please try again.")
72     if __name__ == "__main__":
73         main()
74 # This program implements a simple library management system using classes, loops, and conditional statements.

```

OUTPUT:

```
1 #Write a python code for library management system using classes.loops and conditional statements with inout
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/Or

Library Management System
1. Add Book
2. Display Books
3. Borrow Book
4. Return Book
5. Exit
Enter your choice: 1
Enter book title: The Time Machine
Enter book author: H.G.Wells
Book 'The Time Machine' added to the library.

Library Management System
1. Add Book
2. Display Books
3. Borrow Book
4. Return Book
5. Exit
Enter your choice: 2
1. The Time Machine by H.G.Wells - Available

Library Management System
1. Add Book
2. Display Books
3. Borrow Book
4. Return Book
5. Exit
Enter your choice: 3
1. The Time Machine by H.G.Wells - Available
Enter the book index to borrow: 1
You have borrowed 'The Time Machine'.

Library Management System
1. Add Book
2. Display Books
3. Borrow Book
4. Return Book
5. Exit
Enter your choice: 4
1. The Time Machine by H.G.Wells - Borrowed
Enter the book index to return: 1
You have returned 'The Time Machine'.

Library Management System
1. Add Book
2. Display Books
3. Borrow Book
4. Return Book
5. Exit
Enter your choice: 5
Exiting the system. Goodbye!
○ PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

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.”

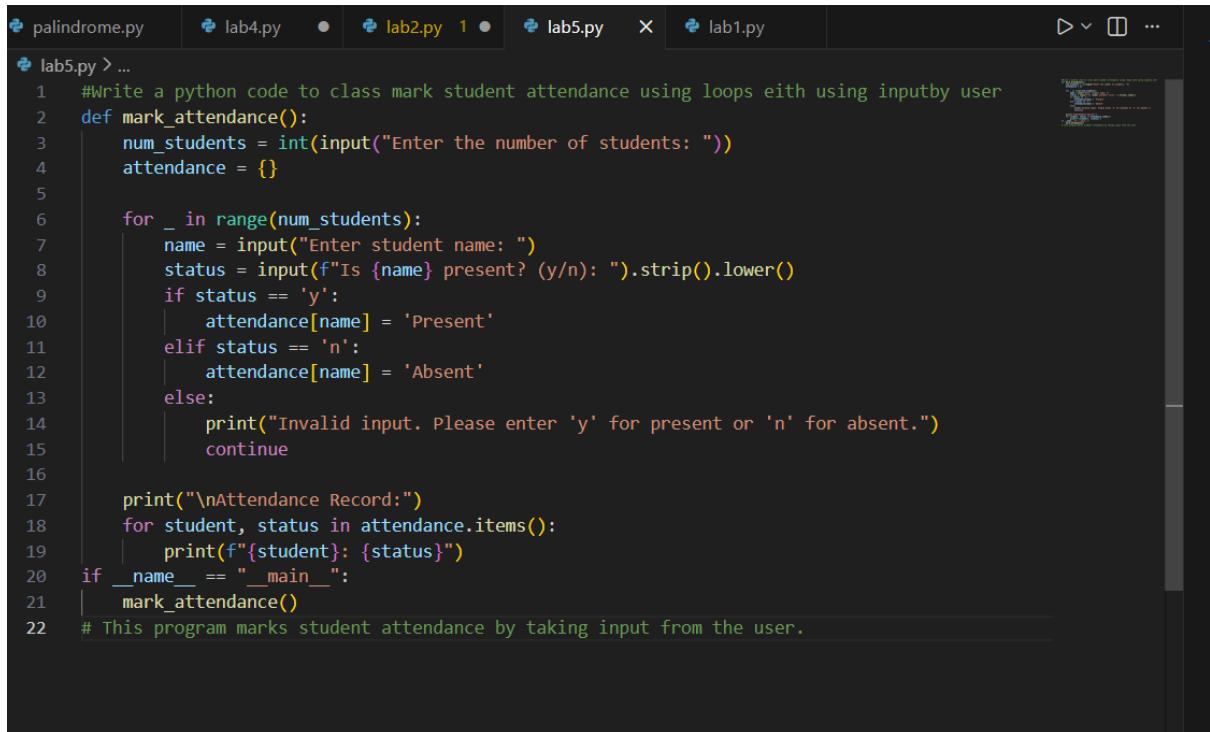
Expected Output:

- AI-generated attendance logic.
- Correct display of attendance.
- Test cases.

PROMPT:

#Write a python code to class mark student attendance using loops eith using inputby user

CODE:



```

palindrome.py | lab4.py | lab2.py | lab5.py | lab1.py
lab5.py > ...
1 #write a python code to class mark student attendance using loops eith using inputby user
2 def mark_attendance():
3     num_students = int(input("Enter the number of students: "))
4     attendance = {}
5
6     for _ in range(num_students):
7         name = input("Enter student name: ")
8         status = input(f"Is {name} present? (y/n): ").strip().lower()
9         if status == 'y':
10             attendance[name] = 'Present'
11         elif status == 'n':
12             attendance[name] = 'Absent'
13         else:
14             print("Invalid input. Please enter 'y' for present or 'n' for absent.")
15             continue
16
17     print("\nAttendance Record:")
18     for student, status in attendance.items():
19         print(f"{student}: {status}")
20 if __name__ == "__main__":
21     mark_attendance()
22 # This program marks student attendance by taking input from the user.

```

OUTPUT:

```
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & c:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab5.py
Enter the number of students: 6
Enter student name: pranathi
Is pranathi present? (y/n): y
Enter student name: lasya
Is lasya present? (y/n): n
Enter student name: milky
Is milky present? (y/n): y
Enter student name: lucky
Is lucky present? (y/n): n
Enter student name: ramya
Is ramya present? (y/n): y
Enter student name: prathibha
Is prathibha present? (y/n): y

Attendance Record:
pranathi: Present
lasya: Absent
milky: Present
lucky: Absent
ramya: Present
lucky: Absent
○ ramya: Present
prathibha: Present
PS C:\Users\thota\OneDrive\Desktop\AIAC>
```

Ln 22, Col 71 S

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.

PROMPT:

#Write a python code to for ATM menu by using loops and conditional statements

CODE:

```
palindrome.py    lab4.py    lab2.py 1    lab5.py    lab1.py
lab5.py > ...
1  #Write a python code to for ATM menu by using loops and conditional statements
2  def atm_menu():
3      balance = 1000 # Initial balance
4      while True:
5          print("\nWelcome to the ATM")
6          print("1. Check Balance")
7          print("2. Deposit Money")
8          print("3. Withdraw Money")
9          print("4. Exit")
10         choice = input("Please select an option (1-4): ")
11         if choice == '1':
12             print(f"Your current balance is: ${balance}")
13         elif choice == '2':
14             amount = float(input("Enter amount to deposit: $"))
15             if amount > 0:
16                 balance += amount
17                 print(f"${amount} deposited successfully.")
18             else:
19                 print("Invalid amount. Please enter a positive number.")
20
21         elif choice == '3':
22             amount = float(input("Enter amount to withdraw: $"))
23             if 0 < amount <= balance:
24                 balance -= amount
25                 print(f"${amount} withdrawn successfully.")
26             else:
27                 print("Invalid amount. Please check your balance and try again.")
28         elif choice == '4':
29             print("Thank you for using the ATM. Goodbye!")
30             break
31         else:
32             print("Invalid selection. Please choose a valid option.")
33     if __name__ == "__main__":
34         atm_menu()
35 ↵ # This program simulates an ATM menu using loops and conditional statements.1
```

OUTPUT:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/Desktop/ATM.py

...
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/Desktop/ATM.py

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 1
Your current balance is: $1000

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 2
Enter amount to deposit: $1000
$1000.0 deposited successfully.

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 1
Your current balance is: $2000.0

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 3
Enter amount to withdraw: $1000
$1000.0 withdrawn successfully.

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 1
Your current balance is: $1000.0

Welcome to the ATM
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 4
3. Withdraw Money
4. Exit
Please select an option (1-4): 4
4. Exit
Please select an option (1-4): 4
Thank you for using the ATM. Goodbye!
PS C:\Users\thota\OneDrive\Desktop\AIAC> 
```

