

AI Assisted Coding Assignment - 6.5

Name: N. Varshith

Ht.no: 2303A51441

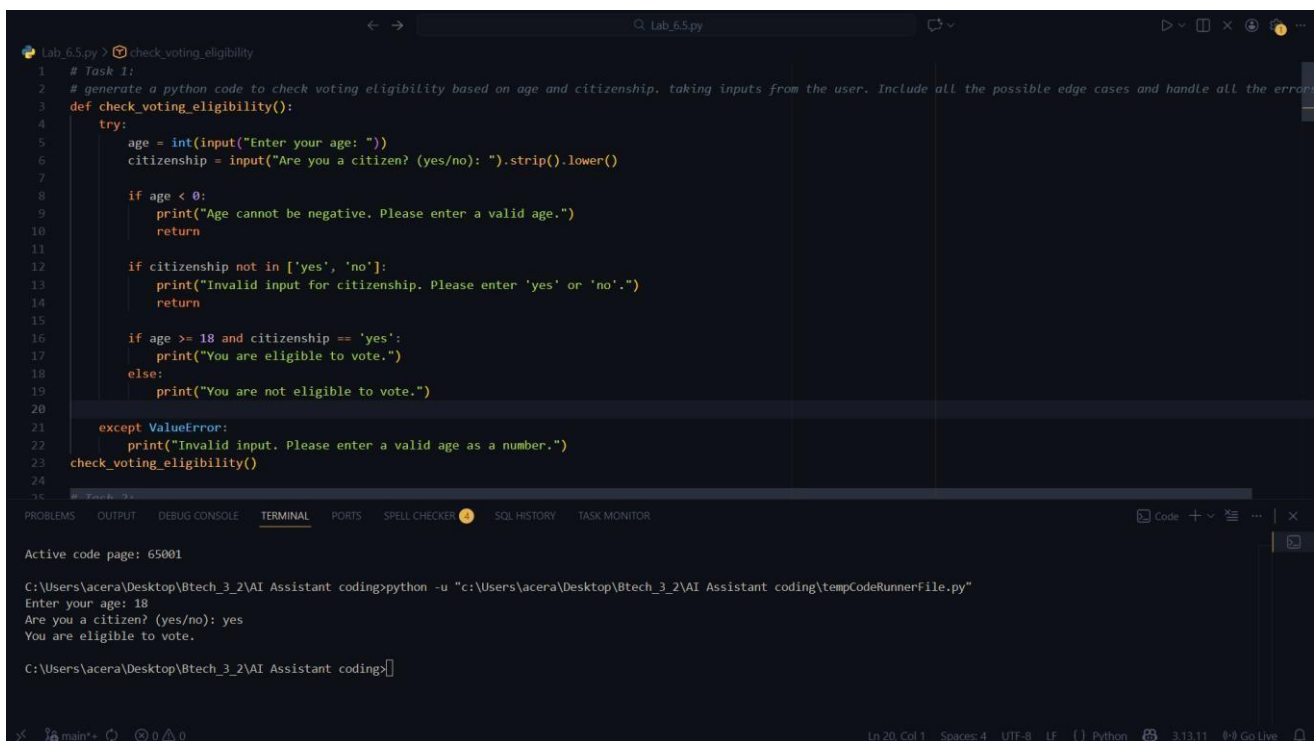
Bt.no: 21

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." Code:



```
Lab_6.5.py > check_voting_eligibility
1 # Task 1:
2 # generate a python code to check voting eligibility based on age and citizenship. taking inputs from the user. Include all the possible edge cases and handle all the errors.
3 def check_voting_eligibility():
4     try:
5         age = int(input("Enter your age: "))
6         citizenship = input("Are you a citizen? (yes/no): ").strip().lower()
7
8         if age < 0:
9             print("Age cannot be negative. Please enter a valid age.")
10            return
11
12            if citizenship not in ['yes', 'no']:
13                print("Invalid input for citizenship. Please enter 'yes' or 'no'.")
14                return
15
16                if age >= 18 and citizenship == 'yes':
17                    print("You are eligible to vote.")
18                else:
19                    print("You are not eligible to vote.")
20
21            except ValueError:
22                print("Invalid input. Please enter a valid age as a number.")
23            check_voting_eligibility()
24
25
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER SQL HISTORY TASK MONITOR
Active code page: 65001
C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>python -u "c:\Users\acera\Desktop\Btech_3_2\AI Assistant coding\tempCodeRunnerFile.py"
Enter your age: 18
Are you a citizen? (yes/no): yes
You are eligible to vote.
C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>
```

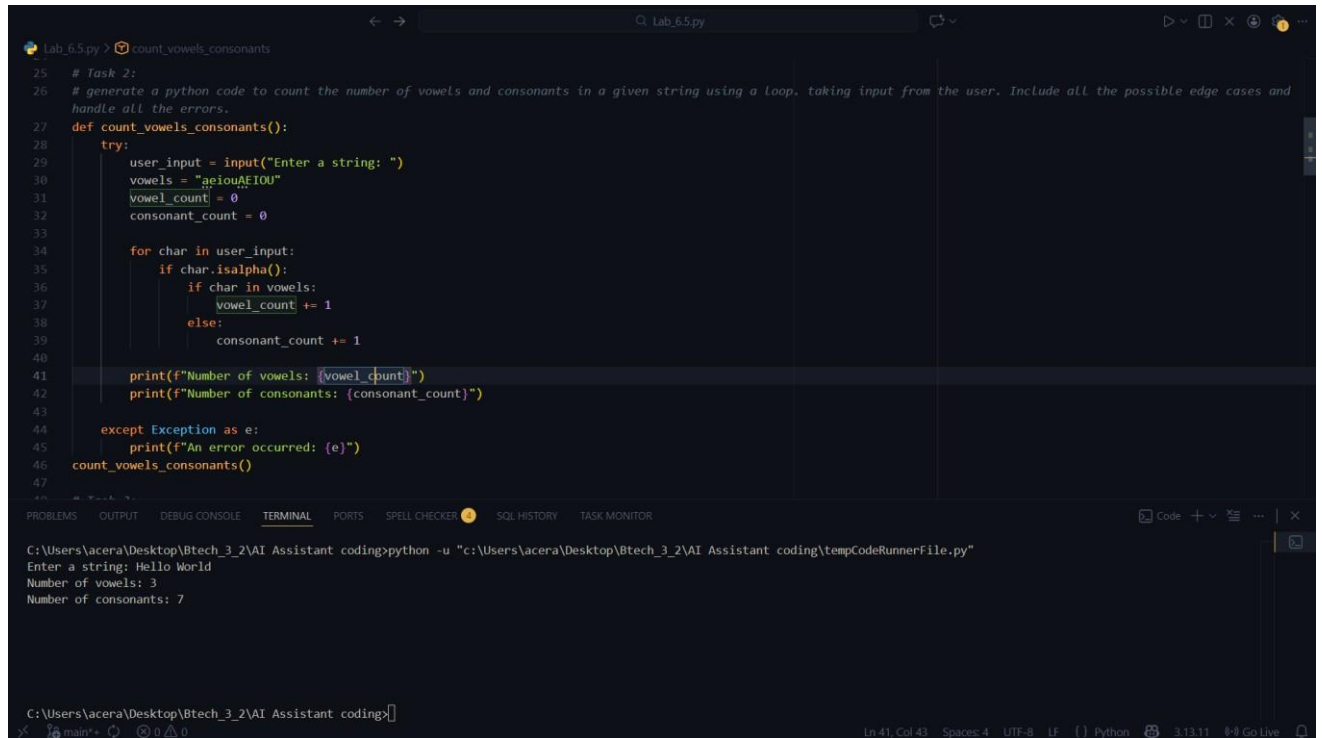
Task Description #2

(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.” **Code:**



```
Lab_6.5.py > count_vowels_consonants
25 # Task 2:
26 # generate a python code to count the number of vowels and consonants in a given string using a loop. taking input from the user. Include all the possible edge cases and
   handle all the errors.
27 def count_vowels_consonants():
28     try:
29         user_input = input("Enter a string: ")
30         vowels = "aeiouAEIOU"
31         vowel_count = 0
32         consonant_count = 0
33
34         for char in user_input:
35             if char.isalpha():
36                 if char in vowels:
37                     vowel_count += 1
38                 else:
39                     consonant_count += 1
40
41         print(f"Number of vowels: {vowel_count}")
42         print(f"Number of consonants: {consonant_count}")
43
44     except Exception as e:
45         print(f"An error occurred: {e}")
46 count_vowels_consonants()
47
```

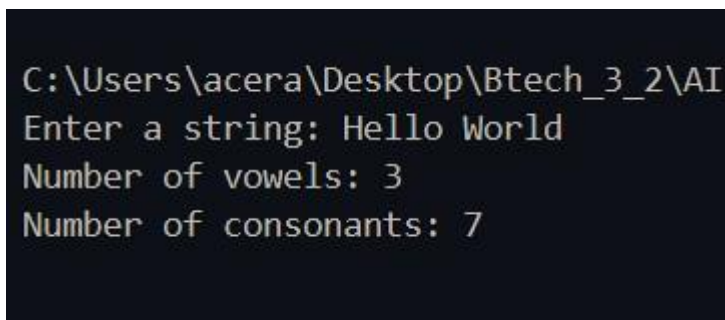
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER SQL HISTORY TASK MONITOR

```
C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>python -u "c:\Users\acera\Desktop\Btech_3_2\AI Assistant coding\tempCodeRunnerFile.py"
Enter a string: Hello World
Number of vowels: 3
Number of consonants: 7

C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>
```

Ln 41, Col 43 Spaces: 4 UTF-8 LF Python 3.13.11 Go Live

Output:



```
C:\Users\acera\Desktop\Btech_3_2\AI
Enter a string: Hello World
Number of vowels: 3
Number of consonants: 7
```

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. Code:

```

Lab_6.5.py > count_vowels_consonants
47
48 # Task 3:
49 # generate a high level python program for a library management system using classes , loops and conditional statements. Try to include all the possible
   edge cases and handle all the errors.
50 class Book:
51     def __init__(self, title, author):
52         self.title = title
53         self.author = author
54         self.is_borrowed = False
55 class Library:
56     def __init__(self):
57         self.books = []
58
59     def add_book(self, book):
60         self.books.append(book)
61         print(f'Book "{book.title}" by {book.author} added to the library.')
62
63     def borrow_book(self, title):
64         for book in self.books:
65             if book.title == title:
66                 if not book.is_borrowed:
67                     book.is_borrowed = True
68                     print(f'You have borrowed "{book.title}"')
69                     return
70                 else:
71                     print(f'Sorry, "{book.title}" is already borrowed.')
72                     return
73         print(f'Sorry, the book "{title}" is not available in the library.')
74
75     def return_book(self, title):
76         for book in self.books:
77             if book.title == title:
78                 if book.is_borrowed:
79                     book.is_borrowed = False
80                     print(f'You have returned "{book.title}"')
81                     return
82
83
84
85 class Library:
86     def __init__(self):
87         print(f'Sorry, the book "{title}" does not belong to this library.')
88
89 def main():
90     library = Library()
91     while True:
92         print("\nLibrary Management System")
93         print("1. Add Book")
94         print("2. Borrow Book")
95         print("3. Return Book")
96         print("4. Exit")
97         choice = input("Enter your choice (1-4): ")
98
99         if choice == '1':
100             title = input("Enter book title: ")
101             author = input("Enter book author: ")
102             book = Book(title, author)
103             library.add_book(book)
104         elif choice == '2':
105             title = input("Enter the title of the book to borrow: ")
106             library.borrow_book(title)
107         elif choice == '3':
108             title = input("Enter the title of the book to return: ")
109             library.return_book(title)
110         elif choice == '4':
111             print("Exiting the Library Management System.")
112             break
113         else:
114             print("Invalid choice. Please enter a number between 1 and 4.")
115
116 if __name__ == "__main__":
117     main()
118
119
120

```

Output:

```

C:\Users\acera\Desktop\Btech_3_2\AI Assistant coding>python -u "c:\Users\acera\Desktop\Btech_3_2\AI Assistant coding\tempCodeRunnerFile.py"

Library Management System
1. Add Book
2. Borrow Book
3. Return Book
4. Exit
Enter your choice (1-4): 1
Enter book title: Tales
Enter book author: Hercules
Book "Tales" by Hercules added to the library.

Library Management System
1. Add Book
2. Borrow Book
3. Return Book
4. Exit
Enter your choice (1-4): 4
Exiting the Library Management System.

```

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.

Code:

```
Lab_6.5.py > Attendance
116 # Task 4:
117 # generate a python class to mark and display attendance of the students using Loops and conditional statements. Include all the possible edge cases and
    handle all the errors.
118 class Attendance:
119     def __init__(self):
120         self.attendance_record = {}
121
122     def mark_attendance(self, student_name, status):
123         if status.lower() not in ['present', 'absent']:
124             print("Invalid status. Please enter 'present' or 'absent'.")
125             return
126         self.attendance_record[student_name] = status.lower()
127         print(f"Attendance marked for {student_name} as {status}.")
128
129     def display_attendance(self):
130         if not self.attendance_record:
131             print("No attendance records available.")
132             return
133         print("\nAttendance Record:")
134         for student, status in self.attendance_record.items():
135             print(f"{student}: {status.capitalize()}")
136
137 def main():
138     attendance = Attendance()
139     while True:
140         print("\nAttendance Management System")
141         print("1. Mark Attendance")
142         print("2. Display Attendance")
143         print("3. Exit")
144         choice = input("Enter your choice (1-3): ")
145
146         if choice == '1':
147             student_name = input("Enter student name: ")
148             status = input("Enter attendance status (present/absent): ")
149             attendance.mark_attendance(student_name, status)
150         elif choice == '2':
151             attendance.display_attendance()
152         elif choice == '3':
153             break
154
155 if __name__ == '__main__':
156     main()
```

```
Lab 6.5.py > Attendance
class Attendance:
    def display_attendance(self):
        for student, status in self.attendance_record.items():
            print(f"{student}: {status.capitalize()}")

def main():
    attendance = Attendance()
    while True:
        print("\nAttendance Management System")
        print("1. Mark Attendance")
        print("2. Display Attendance")
        print("3. Exit")
        choice = input("Enter your choice (1-3): ")

        if choice == '1':
            student_name = input("Enter student name: ")
            status = input("Enter attendance status (present/absent): ")
            attendance.mark_attendance(student_name, status)
        elif choice == '2':
            attendance.display_attendance()
        elif choice == '3':
            print("Exiting the Attendance Management System.")
            break
        else:
            print("Invalid choice. Please enter a number between 1 and 3.")

if __name__ == "__main__":
    main()
```

Ln 118, Col 18 Spaces: 4 UTF-8 LF Python 3.13.11 Go Live

Output:

```
Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 1
Enter student name: sharan
Enter attendance status (present/absent): present
Attendance marked for sharan as present.

Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 2

Attendance Record:
sharan: Present

Attendance Management System
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice (1-3): 3
Exiting the Attendance Management System.
```

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 Code:


```
Lab_6.5.py > atm_menu
161 # Task 5:
162 # Generate a python program using Loops and conditional statements to simulate an ATM menu.
163 # Expected output :
164 # AI-generated menu logic for ATM menu:
165 # correct option handling.
166 # output verification.
167 def atm_menu():
168     balance = 1000 # Initial balance
169     while True:
170         print("\nATM Menu")
171         print("1. Check Balance")
172         print("2. Deposit Money")
173         print("3. Withdraw Money")
174         print("4. Exit")
175         choice = input("Enter your choice (1-4): ")
176
177         if choice == '1':
178             print(f"Your current balance is: ${balance}")
179         elif choice == '2':
180             try:
181                 amount = float(input("Enter the amount to deposit: "))
182                 if amount <= 0:
183                     print("Please enter a positive amount.")
184                 else:
185                     balance += amount
186                     print(f"You have deposited ${amount}. New balance: ${balance}")
187             except ValueError:
188                 print("Invalid input. Please enter a valid number.")
189         elif choice == '3':
190             try:
191                 amount = float(input("Enter the amount to withdraw: "))
192                 if amount <= 0:
193                     print("Please enter a positive amount.")
194                 elif amount > balance:
195                     print("Insufficient funds. Please enter a smaller amount.")
196                 else:
197                     balance -= amount
```

```
Lab_6.5.py > atm_menu
167     def atm_menu():
185         balance += amount
186         print(f"You have deposited ${amount}. New balance: ${balance}")
187     except ValueError:
188         print("Invalid input. Please enter a valid number.")
189     elif choice == '3':
190         try:
191             amount = float(input("Enter the amount to withdraw: "))
192             if amount <= 0:
193                 print("Please enter a positive amount.")
194             elif amount > balance:
195                 print("Insufficient funds. Please enter a smaller amount.")
196             else:
197                 balance -= amount
198                 print(f"You have withdrawn ${amount}. New balance: ${balance}")
199             except ValueError:
200                 print("Invalid input. Please enter a valid number.")
201     elif choice == '4':
202         print("Thank you for using the ATM. Goodbye!")
203         break
204     else:
205         print("Invalid choice. Please enter a number between 1 and 4.")
206 atm_menu()
207
```

Output:

```
Lab 6.5.py > atm menu
167 def atm_menu():
185     balance += amount
186     print(f"You have deposited ${amount}. New balance: ${balance}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER SQL HISTORY TASK MONITOR
Code + - - - X

ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 2
Enter the amount to deposit: 100000
You have deposited $100000.0. New balance: $101000.0

ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 3
Enter the amount to withdraw: 25000
You have withdrawn $25000.0. New balance: $76000.0

ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 1
Your current balance is: $76000.0

ATM Menu
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice (1-4): 4
Thank you for using the ATM. Goodbye!
```

Ln 175, Col 52 Spaces: 4 UTF-8 LF Python 3.13.11 Go Live