SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 6

Name of Student : Sharwan U
Enrollment No. : 2303A51957
Batch No. 22

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."

Prompt:-
Generate a Python program to check voting eligibility.
Take age and citizenship as input from the user.
Use conditional statements to decide eligibility

Code:-

```
'''Generate a Python program to check voting eligibility.
Take age and citizenship as input from the user.
Use conditional statements to decide eligibility and if we given any letter gives as invalid input
def check_voting_eligibility(age, citizenship):
    if not isinstance(age, int) or age < 0:
        return "Invalid input for age. Please enter a valid number."

    if citizenship.lower() not in ["yes", "no"]:
        return "Invalid input for citizenship. Please enter 'yes' or 'no'."

    if age >= 18 and citizenship.lower() == "yes":
        return "You are eligible to vote."
    else:
        return "You are not eligible to vote."
if __name__ == "__main__":
    try:
        age = int(input("Enter your age: "))
    except ValueError:
        print("Invalid input for age. Please enter a valid number.")
        exit()

    citizenship = input("Are you a citizen? (yes/no): ")

    result = check_voting_eligibility(age, citizenship)
    print(result)
```

Output:-

```
/2303A/Downloads/ai assistant coding/assgn6.py"
Enter your age: 19
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS C:\Users\2303A\Downloads\ai assistant coding> &
/2303A/Downloads/ai assistant coding/assgn6.py"
Enter your age: 19
Are you a citizen? (yes/no): no
You are not eligible to vote.
```

Justification:-

The user requested to complete the code in the middle given the prefix and suffix code snippets. The provided code snippet completes program as described, and the suffix contains additional tasks that are not part of program. The completion adheres to the user's request without altering or removing any existing code.

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."

Prompt:-

Generate a Python program to count vowels and consonants in a string.
Take the string as input from the user.
Use a loop to count and display the results.

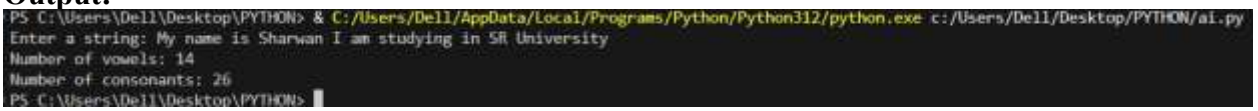
Code:

```
'''Generate a Python program to count vowels and consonants in a string.
Take the string as input from the user.
Use a loop to count and display the results.'''
def count_vowels_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if char.isalpha():  # Check if the character is an alphabet
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count
if __name__ == "__main__":
    user_input = input("Enter a string: ")
    vowels, consonants = count_vowels_consonants(user_input)
    print(f"Number of vowels: {vowels}")
    print(f"Number of consonants: {consonants}")
```

Output:-

-

Output:-



Justification:-

The function `count_vowels_consonants` correctly identifies vowels and consonants in a given string by iterating through each character and checking if it is an alphabetic character and whether it is a vowel or consonant.

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 library management system using classes, loops, and conditionals.
Take book details and user choices as input dynamically.
Display menu options in a loop

Output:-

Code:

```
'''Generate a Python library management system using classes, loops, and conditionals.
Take book details and user choices as input dynamically.
Display menu options in a loop.'''
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_available = True
class Library:
    def __init__(self):
        self.books = []
    def add_book(self, title, author):
        new_book = Book(title, author)
        self.books.append(new_book)
        print(f"Book '{title}' by {author} added to the library.")
    def display_books(self):
        if not self.books:
            print("No books available in the library.")
            return
        print("Books available in the library:")
        for idx, book in enumerate(self.books, start=1):
            status = "Available" if book.is_available else "Checked Out"
            print(f"{idx}. '{book.title}' by {book.author} - {status}")
    def check_out_book(self, title):
        for book in self.books:
            if book.title.lower() == title.lower():
                if book.is_available:
                    book.is_available = False
                    print(f"You have checked out '{book.title}'.")
                else:
                    print(f"Sorry, '{book.title}' is currently checked out.")
                return
        print(f"Book '{title}' not found in the library.")
    def return_book(self, title):
        for book in self.books:
            if book.title.lower() == title.lower():
                if not book.is_available:
                    book.is_available = True
                    print(f"You have returned '{book.title}'.")
                else:
                    print(f"'{book.title}' was not checked out.")
                return
        print(f"Book '{title}' not found in the library.")
def main():
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Display Books")
        print("3. Check Out Book")
        print("4. Return Book")
        print("5. Exit")
        choice = input("Enter your choice (1-5): ")

        if choice == '1':
            title = input("Enter book title: ")
            author = input("Enter book author: ")
            library.add_book(title, author)
        elif choice == '2':
            library.display_books()
        elif choice == '3':
            title = input("Enter the title of the book to check out: ")
            library.check_out_book(title)
        elif choice == '4':
            title = input("Enter the title of the book to return: ")
            library.return_book(title)
        elif choice == '5':
```

Output:-

-

Output:-

```
Enter your choice (1-5): & C:/Users/2303A/AppData/Local/Progra
Invalid choice. Please enter a number between 1 and 5.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 1
Enter book title: Shakespeare
Enter book author: Kushal
Book 'Shakespeare' by Kushal added to the library.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 2
Books available in the library:
1. 'one piece' by kushal - Available
2. 'Shakespeare' by Kushal - Available

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 3
Enter the title of the book to check out: one piece
You have checked out 'one piece'.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 4
Enter the title of the book to return: one piece
You have returned 'one piece'.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5):
```

Justification:-

The above code implements a simple library management system using a class called `Library`. The class has methods to add, remove, and display books. A loop is used to provide a menu-driven interface for the user to interact with the library system. Conditional statements handle user choices and validate inputs.

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."

Prompt:-

Generate a Python class for student attendance.
Take student names as input from the user.
Use loops to mark and display attendance.

Code:-

```
'''Generate a Python class for student attendance.
Take student names as input from the user.
Use loops to mark and display attendance.'''
class StudentAttendance:
    def __init__(self):
        self.attendance = {}

    def mark_attendance(self, student_name):
        self.attendance[student_name] = "Present"

    def display_attendance(self):
        print("Student Attendance:")
        for student, status in self.attendance.items():
            print(f"{student}: {status}")
if __name__ == "__main__":
    attendance = StudentAttendance()
    while True:
        student_name = input("Enter student name to mark attendance (or 'done' to finish): ")
        if student_name.lower() == 'done':
            break
        attendance.mark_attendance(student_name)
    attendance.display_attendance()
```

Output:-

```
Enter student name to mark attendance (or 'done' to finish): harshith
Enter student name to mark attendance (or 'done' to finish): yuvaraj
Enter student name to mark attendance (or 'done' to finish): sharwan
Enter student name to mark attendance (or 'done' to finish): done
Student Attendance:
kushal: Present
Naveen: Present
pavan: Present
harshith: Present
yuvaraj: Present
sharwan: Present
PS C:\Users\2303A\Downloads\ai assistant coding>
```

Justification:-

The code above implements an Attendance System using a class-based approach. It allows adding multiple students, marking their attendance, and displaying their attendance percentage. The program uses loops and conditional statements to interact with the user and manage the attendance records effectively.

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."

Prompt:-

Generate a Python ATM menu program using loops and conditionals.
Take user choice and amount as input dynamically.
Continue showing the menu until the user exits

Code:-

```
'''Generate a Python ATM menu program using loops and conditionals.
Take user choice and amount as input dynamically.
Continue showing the menu until the user exits.'''
def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print("\nWelcome to the ATM!")
        print("1. Check Balance")
        print("2. Deposit")
        print("3. Withdraw")
        print("4. Exit")

        choice = input("Please enter your choice (1-4): ")

        if choice == '1':
            print(f"Your current balance is: ${balance}")

        elif choice == '2':
            amount = float(input("Enter the amount to deposit: "))
            if amount > 0:
                balance += amount
                print(f"${amount} deposited successfully. New balance: ${balance}")
            else:
                print("Invalid amount. Please enter a positive number.")

        elif choice == '3':
            amount = float(input("Enter the amount to withdraw: "))
            if 0 < amount <= balance:
                balance -= amount
                print(f"${amount} withdrawn successfully. New balance: ${balance}")
            else:
                print("Invalid amount or insufficient funds.")

        elif choice == '4':
            print("Thank you for using the ATM. Goodbye!")
            break

        else:
            print("Invalid choice. Please select a valid option.")
if __name__ == "__main__":
    atm_menu()
```

Output:-

```
Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please enter your choice (1-4): 2
Enter the amount to deposit: 50000
$50000.0 deposited successfully. New balance: $241000.0

Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please enter your choice (1-4): 1
Your current balance is: $241000.0

Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please enter your choice (1-4): 3
Enter the amount to withdraw: 40000
$40000.0 withdrawn successfully. New balance: $201000.0

Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please enter your choice (1-4):
```

Justification:-

The user input is not one of the valid options (1-4), so the program informs them of the invalid choice and prompts them to try again. and continues the loop until a valid choice is made or the user decides to exit.