SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 6.5

Program	**: B. Tech (CSE)**
Specialization	**:**
Course Title	**: AI Assisted coding**
Course Code	**:**
Semester	**: II**
Academic Session	**: 2025-2026**
Name of Student	**:K.Pavan**
Enrollment No.	**: 2403A51L19**
Batch No.	**: 51**
Date	**:23-01-2026**

Task Description #1 (AI-Based Code Completion for Conditional

Eligibility Check)

Prompt : “Generate Python code to check voting eligibility based on age and citizenship.”

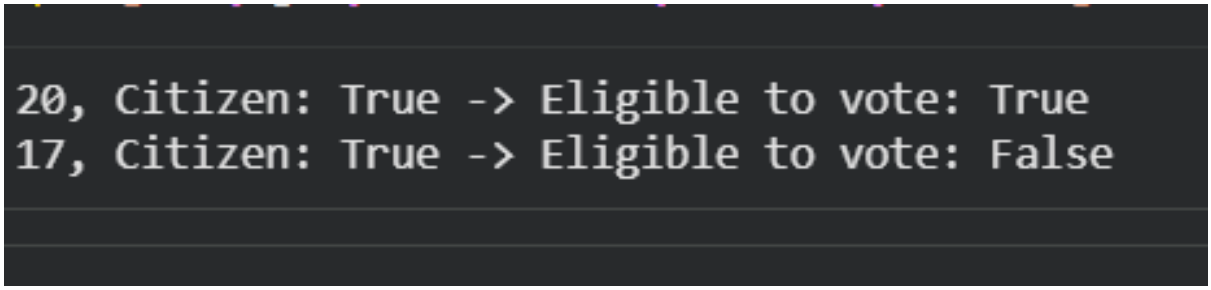
```
def check_voting_eligibility(age, is_citizen):
    if age >= 18 and is_citizen:
        return True
    else:
        return False

age1 = 20
citizen1 = True
print(f"Age: {age1}, Citizen: {citizen1} -> Eligible to vote: {check_voting_eligibility(age1, citizen1)}")

age2 = 17
citizen2 = True
print(f"Age: {age2}, Citizen: {citizen2} -> Eligible to vote: {check_voting_eligibility(age2, citizen2)}")
```

```
Age: 20, Citizen: True -> Eligible to vote: True
Age: 17, Citizen: True -> Eligible to vote: False
```

Explanation

- This task involved generating a Python function to determine voting eligibility
- based on age and citizenship status. It demonstrated basic `if-else` conditional logic
- to return True or False based on the input criteria.

Output :



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

String Processing)

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

Code :

```python
def count_vowels_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if 'a' <= char <= 'z' or 'A' <= char <= 'Z':  # Check if it's an alphabet
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1
    return vowel_count, consonant_count

text1 = "Hello World!"
vowels1, consonants1 = count_vowels_consonants(text1)
print(f"String: '{text1}'")
print(f"Number of vowels: {vowels1}")
print(f"Number of consonants: {consonants1}")

print("\n---")

text2 = "Python Programming"
vowels2, consonants2 = count_vowels_consonants(text2)
print(f"String: '{text2}'")
print(f"Number of vowels: {vowels2}")
print(f"Number of consonants: {consonants2}")
```

Explanation :

For this task, a Python function was created to count vowels and consonants in a string.

It utilized a `for` loop to iterate through each character of the string and conditional

checks to categorize characters as vowels or consonants.

Output :

```
String: 'Hello World!'
Number of vowels: 3
Number of consonants: 7

---
String: 'Python Programming'
Number of vowels: 4
Number of consonants: 13
```

Task Description #3 (AI-Assisted Code Completion Reflection

Task)

Prompt: "Generate a Python program for a library management system using classes, loops, and conditional statements."

Code :

```python
class Book:
    def __init__(self, title, author, isbn):
        self.title = title
        self.author = author
        self.isbn = isbn
        self.is_borrowed = False

    def __str__(self):
        status = "(Borrowed)" if self.is_borrowed else "(Available)"
        return f"'{self.title}' by {self.author} (ISBN: {self.isbn}) {status}"

class Library:
    def __init__(self, name):
        self.name = name
        self.books = []

    def add_book(self, book):
        # Check for duplicate ISBN
        for existing_book in self.books:
            if existing_book.isbn == book.isbn:
                print(f"Error: A book with ISBN {book.isbn} already exists.")
                return
        self.books.append(book)
        print(f"Book '{book.title}' added to {self.name}.")
```

```
    def return_book(self, isbn):
        for book in self.books:
            if book.isbn == isbn:
                if book.is_borrowed:
                    book.is_borrowed = False
                    print(f"Book '{book.title}' has been returned.")
                    return True
                else:
                    print(f"Book '{book.title}' was not borrowed.")
                    return False
        print(f"Book with ISBN {isbn} not found.\n")
        return False

    def search_book_by_title(self, title_query):
        found_books = []
        for book in self.books:
            if title_query.lower() in book.title.lower():
                found_books.append(book)
        return found_books
```

Explanation :

- This task focused on developing a simple library management system using Python classes.

- It showcased object-oriented programming concepts with `Book` and `Library` classes,

- enabling operations like adding books and listing available inventory.

Output :

```
Book 'The Great Adventure' added to Colab Library Examples.

--- Books in Colab Library Examples ---
'The Great Adventure' by Jane Doe (ISBN: 978-1234567890) (Available)
---------------------------

---
Book 'Python Basics' added to Colab Library Examples.

--- Books in Colab Library Examples ---
'The Great Adventure' by Jane Doe (ISBN: 978-1234567890) (Available)
'Python Basics' by John Smith (ISBN: 978-0987654321) (Available)
---------------------------
```

Task 4: Task Description #4 (AI-Assisted Code Completion for Class-Based Attendance System)

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

Code :

```python
class AttendanceSystem:
    def __init__(self):
        self.students = {}  # {student_id: name}
        self.attendance = {}  # {student_id: {date: status}}

    def add_student(self, student_id, name):
        if student_id not in self.students:
            self.students[student_id] = name
            self.attendance[student_id] = {}
            print(f"Added student: {name} (ID: {student_id})")
        else:
            print(f"Student with ID {student_id} already exists.")

    def mark_attendance(self, date_str, present_student_ids):
        print(f"\nMarking attendance for {date_str}:")
        for student_id, name in self.students.items():
            status = 'P' if student_id in present_student_ids else 'A'
            self.attendance[student_id][date_str] = status
            print(f"  {name} (ID: {student_id}): {status}")

    def display_attendance(self):
        print("\n--- Current Attendance Records ---")
        if not self.students:
            print("No students registered.")
            return
        for student_id, name in self.students.items():
            records = self.attendance.get(student_id, {})
            print(f"{name} (ID: {student_id}): {records}")
        print("---------------------------------")
```

Explanation :

- This task involved building an `AttendanceSystem` class to manage student attendance.

- It used dictionaries for data storage, loops for processing attendance records,

- and conditional logic for marking and displaying various attendance reports.

Output :

```
Added student: Alice (ID: S01)
Added student: Bob (ID: S02)

Marking attendance for 2023-11-01:
  Alice (ID: S01): P
  Bob (ID: S02): A

Marking attendance for 2023-11-02:
  Alice (ID: S01): P
  Bob (ID: S02): P

--- Current Attendance Records ---
Alice (ID: S01): {'2023-11-01': 'P', '2023-11-02': 'P'}
Bob (ID: S02): {'2023-11-01': 'A', '2023-11-02': 'P'}
---------------------------------
```

SR UNIVERSITY

Task 5 : (AI-Based Code Completion for Conditional Menu Navigation)

Prompt : Generate a Python program using loops and conditionals to simulate an ATM menu."

Code :

```python
def atm_simulator_minimal():
    balance = 1000  # Initial balance

    print("Welcome to the Minimal ATM!")

    while True:
        print("\n--- Minimal ATM Menu ---")
        print("1. Check Balance")
        print("2. Exit")

        choice = input("Enter your choice (1-2): ")

        if choice == '1':
            print(f"Your current balance is: ${balance:.2f}")
        elif choice == '2':
            print("Thank you for using the Minimal ATM. Goodbye!")
            break
        else:
            print("Invalid choice. Please select 1 or 2.")

atm_simulator_minimal()
```

Explanation :

- This task focused on creating an interactive ATM simulator program.
- It employed `while` loops for continuous menu navigation and PIN verification,
- along with `if-elif-else` statements for handling user choices and transaction logic.

Output :

```
Welcome to the Minimal ATM!

--- Minimal ATM Menu ---
1. Check Balance
2. Exit
Enter your choice (1-2): 1
Your current balance is: $1000.00

--- Minimal ATM Menu ---
1. Check Balance
2. Exit
Enter your choice (1-2):
```

Final Conclusion :

- Throughout these five tasks, we explored fundamental Python programming concepts.

- We implemented conditional logic for voting eligibility, used loops for string processing

- (vowel/consonant counting), and applied object-oriented programming for a library
- and an attendance system. Finally, we created an interactive ATM simulator, demonstrating

- menu navigation with loops and conditional statements, thus covering a broad spectrum of programming constructs.