

## School of Computer Science and Artificial Intelligence

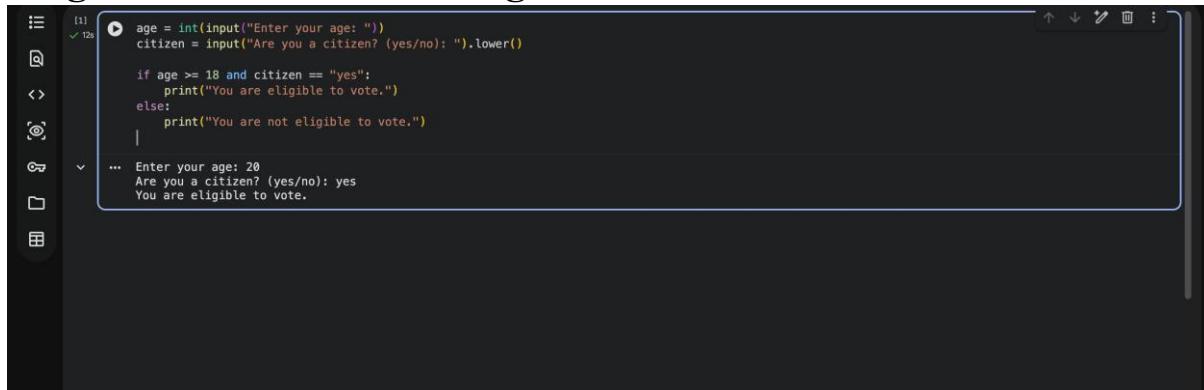
### Lab Assignment # 6.5

Program : B. Tech (CSE)  
Specialization :  
Course Title : AI Assisted coding  
Course Code :  
Semester : II  
Academic Session : 2025-2026  
Name of Student : A.Nagasai  
Enrollment No. : 2403A51L18  
Batch No. : 51  
Date : 23-01-2026

### Task 1: Use an AI tool to generate eligibility logic.

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

### AI-generated conditional logic.



```
age = int(input("Enter your age: "))
citizen = input("Are you a citizen? (yes/no): ").lower()

if age >= 18 and citizen == "yes":
    print("You are eligible to vote.")
else:
    print("You are not eligible to vote.")

...
Enter your age: 20
Are you a citizen? (yes/no): yes
You are eligible to vote.
```

A screenshot of a code editor window showing a Python script. The script prompts the user for their age and citizenship status, then uses an if condition with an and logical operator to determine if they are eligible to vote. The code is syntax-highlighted, and the output of the program running in the terminal below it is shown in gray text.

### Explanation of conditions

- The program takes age and citizenship status as input.
- The if condition checks two rules
  - Age must be 18 or above
  - Citizenship must be “yes”
- The logical operator and ensures both conditions must be true.

- If both conditions are satisfied, the user is eligible to vote.
- Otherwise, the program prints not eligible.

### Explanation of conditions.

- Age = 20, Citizen = yes → **Eligible**
- Age = 16, Citizen = yes → **Not Eligible**

## 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.”

### AI-Generated Code

```
▶ text = input("Enter a string: ").lower()
vowels = "aeiou"
vowel_count = 0
consonant_count = 0

for ch in text:
    if ch.isalpha():
        if ch in vowels:
            vowel_count += 1
        else:
            consonant_count += 1

print("Vowels:", vowel_count)
print("Consonants:", consonant_count)
```

### Correct counts.

- Converts input to lowercase
- Uses for loop to read each character
- Checks: Alphabet → vowel or consonant
- Counts correctly

### Output verification.

```
Enter a string: ROHAN
Vowels: 2
Consonants: 3
```

## 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.”

### AI-generated conditional logic.

```
▶ class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)

    def display_books(self):
        if not self.books:
            print("No books available.")
        else:
            for book in self.books:
                print(book)

lib = Library()

while True:
    print("1. Add Book")
    print("2. Display Books")
    print("3. Exit")

    choice = int(input("Enter choice: "))

    if choice == 1:
        book = input("Enter book name: ")
        lib.add_book(book)
    elif choice == 2:
        lib.display_books()
    elif choice == 3:
        break
    else:
        print("Invalid choice")
```

## Review of AI suggestions quality.

- The AI correctly generated:
  - A class (`Library`)
  - Loop (`while True`) for menu repetition
  - Conditional statements (`if-elif-else`) for user choices
- Code is easy to understand and readable ● Uses basic data structures (list) efficiently
  - Minor improvement needed:
  - Input validation can be added.
  - Exception handling for invalid inputs.

## Short reflection on AI-assisted coding experience.

AI-assisted code completion helped in quickly generating a structured and functional program. It reduced development time and provided a clear logical flow using classes, loops, and conditionals. However, reviewing and improving the AI-generated code is necessary to ensure correctness, efficiency, and better error handling. AI should be used responsibly as a support tool while maintaining strong programming fundamentals.

## 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.”

## AI-generated attendance logic.

```

▶ class Attendance:
    def __init__(self):
        self.students = {}

    def mark_attendance(self, name, status):
        self.students[name] = status

    def display_attendance(self):
        for name, status in self.students.items():
            print(name, ":", status)

# ----- Test Case 1 -----
att = Attendance()
att.mark_attendance("Alice", "Present")
att.mark_attendance("Bob", "Absent")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Absent

# ----- Test Case 2 -----
att.mark_attendance("Charlie", "Present")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Absent
# Charlie : Present

# ----- Test Case 3 -----
att.mark_attendance("Bob", "Present")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Present
# Charlie : Present

```

## Correct display of attendance.

```

▶ class Attendance:
    def __init__(self):
        self.students = {}

    def mark_attendance(self, name, status):
        self.students[name] = status

    def display_attendance(self):
        for name in sorted(self.students):
            print(name, ":", self.students[name])

# Test Case
att = Attendance()
att.mark_attendance("Alice", "Present")
att.mark_attendance("Bob", "Absent")
att.mark_attendance("Charlie", "Present")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Absent
# Charlie : Present

... Alice : Present
    Bob : Absent
    Charlie : Present

```

## Testcases.

```
# ----- Test Case 1 -----
att = Attendance()
att.mark_attendance("Alice", "Present")
att.mark_attendance("Bob", "Absent")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Absent

# ----- Test Case 2 -----
att.mark_attendance("Charlie", "Present")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Absent
# Charlie : Present

# ----- Test Case 3 -----
att.mark_attendance("Bob", "Present")
att.display_attendance()

# Expected Output:
# Alice : Present
# Bob : Present
# Charlie : Present

Alice : Present
Bob : Absent
Alice : Present
Bob : Absent
Charlie : Present
Alice : Present
Bob : Present
Charlie : Present
```

## 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.” **AI-generated attendance logic.**

```
balance = 1000

while True:
    print("\n1. Check Balance")
    print("2. Deposit")
    print("3. Withdraw")
    print("4. Exit")

    choice = int(input("Enter your choice: "))

    if choice == 1:
        print("Balance:", balance)

    elif choice == 2:
        amount = int(input("Enter deposit amount: "))
        balance += amount
        print("Amount deposited successfully")

    elif choice == 3:
        amount = int(input("Enter withdraw amount: "))
        if amount <= balance:
            balance -= amount
            ..... print("Please collect your cash")
        else:
            print("Insufficient balance")

    elif choice == 4:
        print("Thank you for using ATM")
        break

    else:
        print("Invalid option")
```

## Correct option handling.

1. Option 1 → Displays balance

2. Option 2 → Deposits amount
3. Option 3 → Withdraws if balance is sufficient
4. Option 4 → Exits menu
5. Invalid input → Error message

## Output verification.

```
..  
1. Check Balance  
2. Deposit  
3. Withdraw  
4. Exit  
Enter your choice: 2  
Enter deposit amount: 2000  
Amount deposited successfully  
  
1. Check Balance  
2. Deposit  
3. Withdraw  
4. Exit  
Enter your choice: 1  
Balance: 3000  
  
1. Check Balance  
2. Deposit  
3. Withdraw  
4. Exit  
Enter your choice: 3  
Enter withdraw amount: 3000  
Please collect your cash  
  
1. Check Balance  
2. Deposit  
3. Withdraw  
4. Exit  
Enter your choice: 1  
Balance: 0  
  
1. Check Balance  
2. Deposit  
3. Withdraw  
4. Exit  
Enter your choice: 4  
Thank you for using ATM
```