

School of Computer Science and Artificial Intelligence

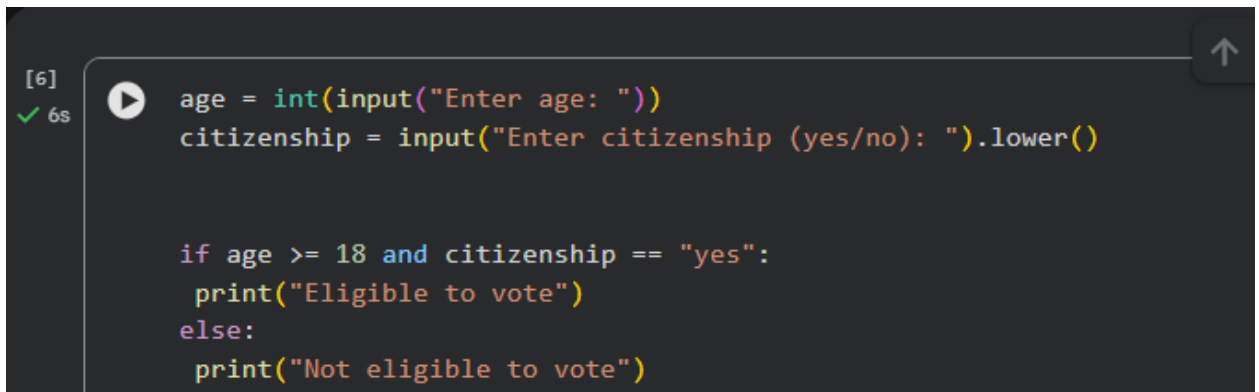
Lab Assignment # 6.5

Program	: B. Tech (CSE)
Specialization	:
Course Title	: AI Assisted coding
Course Code	:
Semester	: II
Academic Session	: 2025-2026
Name of Student	: M.Srinath
Enrollment No.	: 2403A51L29
Batch No.	: 51
Date	:23-01-2026

Task-1

Prompt

Generate Python code to check voting eligibility based on age and citizenship.

A screenshot of a code editor with a dark background. On the left, there is a status bar showing "[6]" and a green checkmark with "6s". A play button icon is next to the first line of code. The code is as follows:

```
age = int(input("Enter age: "))
citizenship = input("Enter citizenship (yes/no): ").lower()

if age >= 18 and citizenship == "yes":
    print("Eligible to vote")
else:
    print("Not eligible to vote")
```

An upward arrow icon is in the top right corner of the editor.

Explanation:

`age = int(input(...))` takes the user's age as an integer.

`citizenship = input(...).lower()` reads citizenship status and converts it to lowercase for comparison.

`if age >= 18 and citizenship == "yes":` checks both conditions together.

`print("Eligible to vote")` executes when both conditions are true.

`else:` handles cases where one or both conditions fail.

Output:

```
print("Not eligible to vote")
```

▼
Enter age: 18
Enter citizenship (yes/no): yes
Eligible to vote

Task-2

Prompt:

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

```
[9] ✓ 9s
text = input("Enter a string: ")
vowels = "aeiou"
vowel_count = 0
consonant_count = 0

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

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

Explanation

- `text = input(...)` reads the input string.
- `vowels = "aeiou"` stores all vowel characters.
- **Counters are initialized to zero.**
- `for ch in text.lower():` iterates through each character.
- `ch.isalpha()` ensures only letters are counted.
- **Inner condition checks vowel or consonant.**
- **Final print statements display the counts.**

Output:

```
... Enter a string:  hello world
Vowels: 3
Consonants: 1
```

Task-3

Prompt:

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

```
[16] ✓ 40s 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("\n1. 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")
```

Explanation

- `Library` class stores book details.
- `List books` is used to manage book records.
- `add_book()` adds a new book to the list.
- `display_books()` uses a loop to display all books.
- `while True` loop allows repeated user interaction.
- `if-elif-else` handles menu selection using conditionals.

Output:

```
...
1. Add Book
2. Display Books
3. Exit
Enter choice: 1
Enter book name: python

1. Add Book
2. Display Books
3. Exit
Enter choice: 1
Enter book name: java


1. Add Book
2. Display Books
3. Exit
Enter choice: 2
python
java

1. Add Book
2. Display Books
3. Exit
Enter choice: 3
```

Task-4:

Prompt Used:

Generate a Python class to mark and display student attendance using loops.

```
[18]
✓ 0s  class Attendance:
    def __init__(self):
        self.records = {}

    def mark_attendance(self, student_name, present):
        self.records[student_name] = present

    def display_attendance(self):
        for name, status in self.records.items():
            if status:
                print(name, "- Present")
            else:
                print(name, "- Absent")
```

Explanation

- **Attendance** class stores attendance details.
- **Dictionary** records stores student names and their attendance status.
- **mark_attendance()** records whether a student is present or absent.
- **display_attendance()** uses a loop to display attendance for all students.

Output:

```
[19] ✓ 0s att = Attendance()

att.mark_attendance("Asritha", True)
att.mark_attendance("Rahul", False)
att.mark_attendance("Sneha", True)

att.display_attendance()

... Asritha - Present
    Rahul - Absent
    Sneha - Present
```

Task-5

Prompt:

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

17]
✓ 1m

balance = 5000

```
while True:
    print("\nATM MENU")
    print("1. Check Balance")
    print("2. Deposit")
    print("3. Withdraw")
    print("4. Exit")

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

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

    elif choice == 2:
        amount = int(input("Enter deposit amount: "))
        if amount > 0:
            balance += amount
            print("Amount deposited successfully")
        else:
            print("Invalid amount")

    elif choice == 3:
        amount = int(input("Enter withdrawal amount: "))
        if amount <= balance and amount > 0:
            balance -= amount
            print("Please collect your cash")
        else:
            print("Insufficient balance or invalid amount")

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

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


Explanation

- `while True` creates a continuous menu loop.
- Menu options are displayed using `print`.
- `if-elif-else` handles user selections.
- Balance is updated using conditional checks.
- Loop exits when the user selects option 4.

Output:

```

v ...
ATM MENU
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 1
Current Balance: 5000

ATM MENU
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 2
Enter deposit amount: 5000
Amount deposited successfully

ATM MENU
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 3
Enter withdrawal amount: 2000
Please collect your cash

ATM MENU
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 4
Thank you for using ATM
```

Conclusion

In this experiment, AI-based code completion was used to generate Python programs for all five tasks involving conditionals, loops, classes, and menu-driven logic. Each task helped in understanding how AI can assist in writing correct and structured code. By reviewing outputs and making improvements, the importance of testing and ethical use of AI tools was learned.