AI Assisted Coding (III Year) Assignment

NAME : K.AKSHAY

HT NO :2303A52209

BATCH :35

Experiment 6: AI-Based Code Completion Working with Suggestions for Classes, Loops, and Conditionals Week 3 – Friday

Learning Outcomes LO1: Use AI-based tools to generate Python code using classes, loops, and conditionals.

LO2: Explain AI-generated code line-by-line.

LO3: Identify errors or inefficiencies in AI-generated code.

LO4: Optimize AI-generated code for readability and performance.

LO5: Demonstrate responsible use of AI tools in coding tasks.

Task Description 1: AI-Based Code Completion for Conditional Eligibility Check

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

AI-Generated Code age = int(input("Enter age: ")) citizen = input("Are you a citizen? (yes/no): ")

if age >= 18 and citizen.lower() == "yes": print("Eligible to vote") else: print("Not eligible to vote")

Explanation

Takes age and citizenship as input

Uses conditional statements

Person must be 18 or above and a citizen

Output Verification Enter age: 20 Are you a citizen? yes Eligible to vote Task Description

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

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

```
Enter age: 20
Are you a citizen? (yes/no): yes
Eligible to vote
```

2: AI-Based Code Completion for Loop-Based String Processing

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

AI-Generated Code text = input("Enter a string: ") vowels = "aeiouAEIOU" 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) Explanation Loop iterates through each character

Checks for alphabets only

Counts vowels and consonants separately

Output Verification Enter a string: Hello World Vowels: 3 Consonants: 7

```python
text = input("Enter a string: ")
vowels = "aeiouAEIOU"
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)
```

```
Enter a string: asdfghjk
Vowels: 1
Consonants: 7
```

Task Description 3: AI-Assisted Code Completion Reflection Task

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

AI-Generated Code 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() lib.add_book("Python Basics") lib.add_book("DataStructures") lib.display_books()

Review of AI Suggestions Code is simple and readable

Uses class, loop, and conditional

Suitable for beginners

Reflection AI helped generate a complete working program quickly. Human review is still needed to add features like issue and return of books.

```
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()
lib.add_book("Python Basics")
lib.add_book("Data Structures")
lib.display_books()
```

```
Python Basics
Data Structures
```

Task Description 4: AI-Based Code Completion for Class-Based Attendance System

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

AI-Generated Code class Attendance: def init(self): self.records = {}

def mark_attendance(self, name, status):

self.records[name] = status

def display_attendance(self):

for name, status in self.records.items():

print(name, ":", status)

att = Attendance() att.mark_attendance("Alice", "Present") att.mark_attendance("Bob", "Absent") att.display_attendance()

Output Alice : Present Bob : Absent Observation

The AI-generated class correctly stores and displays attendance using loops.

```
class Attendance:
    def __init__(self):
        self.records = {}

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

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

att = Attendance()
att.mark_attendance("Alice", "Present")
att.mark_attendance("Bob", "Absent")
att.display_attendance()
```

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

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

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

AI-Generated Code balance = 5000

while True: print("1. Check Balance") print("2. Withdraw") print("3. Exit")

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

if choice == 1:

print("Balance:", balance)

elif choice == 2:

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

if amount <= balance:

balance -= amount

print("Withdrawal successful")

else:

print("Insufficient balance")

elif choice == 3:

print("Thank you")

break

else:

print("Invalid option")

Output Verification Enter choice: 1 Balance: 5000

```
while True:
    print("1. Check Balance")
    print("2. Withdraw")
    print("3. Exit")

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

    if choice == 1:
        print("Balance:", balance)
    elif choice == 2:
        amount = int(input("Enter amount: "))
        if amount <= balance:
            balance -= amount
            print("Withdrawal successful")
        else:
            print("Insufficient balance")
    elif choice == 3:
        print("Thank you")
        break
    else:
        print("Invalid option")
```

```
1. Check Balance
2. Withdraw
3. Exit
Enter choice: 1
Balance: 5000
1. Check Balance
2. Withdraw
3. Exit
Enter choice: 3
Thank you
```

Ethical and Responsible Use of AI AI-generated code was reviewed before use

Logic was tested using sample inputs

AI was used as an assistant, not a replacement for learning

Conclusion

This experiment demonstrates how AI-based code completion tools can help generate Python programs involving classes, loops, and conditionals. While AI improves productivity, human understanding and verification are essential to ensure correctness, efficiency, and ethical usage.